1992-05

Composite system analysis of advanced shipboard electrical power distribution systems

Whitcomb, Clifford Alan

Author(s)	Whitcomb, Clifford Alan.
Title	Composite system analysis of advanced shipboard electrical power distribution systems
Publisher	
Issue Date	1992-05
URL	http://hdl.handle.net/10945/24379

COMPOSITE SYSTEM ANALYSIS OF ADVANCED SHIPBOARD ELECTRICAL POWER DISTRIBUTION SYSTEMS

by

CLIFFORD ALAN WHITCOMB

B. S. Engineering, University of Washington
(1984)

Submitted to the Department of
OCEAN ENGINEERING
in Partial Fulfillment of the
Requirements for the Degrees of

NAVAL ENGINEER

and

MASTER OF SCIENCE IN ELECTRICAL ENGINEERING
AND COMPUTER SCIENCE

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

May 1992

Signature of Author: [redacted]
Department of Ocean Engineering, May 1992

Certified by: [redacted]
Professor James L. Kirtley, Jr.
Department of Electrical Engineering and Computer Science
Thesis Supervisor

Certified by: [redacted]
Professor A. Douglas Carmichael
Department of Ocean Engineering
Thesis Reader

Accepted by: [redacted]
A. Douglas Carmichael, Chairman
Departmental Graduate Committee
Department of Ocean Engineering

COMPOSITE SYSTEM ANALYSIS OF ADVANCED SHIPBOARD ELECTRICAL POWER DISTRIBUTION SYSTEMS

by

CLIFFORD ALAN WHITCOMB

Submitted to the Department of Ocean Engineering
on May 8, 1992 in partial fulfillment of the requirments
for the Degrees of Naval Engineer and Master of Science
in Electrical Engineering and Computer Science

Abstract

Survivability improvement techniques such as equipment separation, redundancy, and arrangement form an integral part of the ship design strategy. The development of tools which can perform assessments of survivability features along with the feasibility, benefits, and costs of such features is required. The ability to evaluate systems in the early or conceptual stages of design is most important to provide the highest potential pay off. A methodology is proposed to perform survivability analyses of composite naval shipboard electrical power distribution systems. The methodology allows the system designer to quantify survivability of various system arrangements, architectures, and control rules. The methodology is coded as an additional capability to an existing system reliability and availability analysis program. The additional analysis sections provide new capabilities for the specific investigation of electric distribution system design alternatives. A method to quantify incremental acquisition and combat effectiveness costs of providing improved survivability is provided. Reliability and availability analysis capabilities are presented. A conceptual naval shipboard electrical power distribution systems is analyzed to demonstrate the techniques employed. For the purposes of this thesis, the data input is specific for electrical distribution systems, but any distributed, interconnected system whether it is an electrical, mechanical, or fluid system could be analyzed with this program.

Thesis Advisor: Dr. James L Kirtley, Jr.
Title: Professor of Electrical Engineering and Computer Science

Acknowledgements

The author would like to acknowledge the assistance provided by the many people who helped make this thesis possible. First of all, Professor Kirtley, who not only taught me a better way to think about solving problems, but who also allowed me to delve into an area of study which interested me. My thanks go also to Dr. Marija Ilic of the MIT LEES laboratory for the introduction into the electrical systems control area through her new courses at MIT. The efforts of Professor Carmichael of the Ocean Engineering department are also greatly appreciated.

Many thanks go to the tireless efforts of my Charles Stark Draper Laboratory advisor, Dr. Deborah Allinger, who made this a great learning experience and ensured that this was a useful area of research. Thanks, too, to Andre Schor and Frank Leong of CSDL for invaluable help in the building and debugging of my computer code. Thanks also to John Sweeney, Education Director of CSDL, for introducing me to the research possibilities there.

I would never have made it this far in my education without the inspiration and support from the people around me who shaped my life. My entire family, from my mother, Sophie, to all of my sisters, Janice, Sara, Ellen, and Emilie, give me love and support always, as does my aunt Jinny Warfield. Many thanks to my grandparents, Ralph and Clara Whitcomb for their love and support. I would also like to thank my future wife, Jennifer Davies, for all of her understanding, love, and support while I spent time on my studies. And to my two best friends, Brian Maher and David Willett, thanks for being there always. It was great to be back home to go to school. I would also like to recognize my advisor from the University of Washington, Amy Maki, who always knew I would make it here where I wanted to be, at MIT.

And most of all, thanks to my mentors, my father, Ralph W. Whitcomb, Jr, my uncle Bob Warfield, and Dr. Hurd Willett of MIT. Their inspiration and guidance will be with me always. I only wish they could be here so I could extend them my deepest thanks. They will be a part of me always.

Table of Contents

List of Figures

Chapter 1. Introduction

Survivability improvement techniques such as equipment separation, redundancy, arrangement, and control system architectures form an integral part of the ship design strategy. Tools which can rapidly and conveniently perform assessments of survivability features, including system control, along with the feasibility, benefits, and cost of such features are required. Of particular importance is the capability to evaluate systems in the early, conceptual stages of design even before ship construction specifications are available, since it is at this point of development when the highest potential pay off can be realized.

Objective

The objective of this thesis is to develop a methodology for evaluating conceptual level electrical power distribution system designs for multiple attributes such as survivability, reliability, and availability with the emphasis on the quantification of the system survivability. Included in this methodology is the ability to perform evaluation of potential control strategies for a given system . A secondary objective is to implement this methodology in a computer program. The program should also include the ability to evaluate affordability by quantifying the incremental cost of providing improved survivability, reliability, and availability.

Background

Electrical distribution system reliability analyses are performed by commercial utilities on large, complex, interconnected systems. These analyses result in the determination of reliability indices for given system configurations. The analyses are generally concerned with either system adequacy or security. Adequacy refers to the existence of enough system facilities to provide energy to meet demand and the existence of properly connected distribution system components to supply the generated power to the desired loads. Adequacy is associated with the static conditions which do not include system disturbances. Security deals with the system's transient response to internal disturbances or perturbations such as the loss of generation or transmission facilities. Most of the probabilistic evaluation techniques currently available deal with system adequacy assessment.[Ref 1]

Electrical power distribution system reliability is the probability that the system can provide sufficient power under given operating conditions over time. Survivability can be considered as reliability under large system stress over an extremely short time period. The analysis of the two concepts is therefore similar in nature, and the same methods can be used to evaluate them.

Since reliability and survivability are associated with random events, they are best evaluated using probability theory. Either analytical or simulation methods can be used in system assessments. Analytical techniques, such as Markov modeling, use models to represent the systems and mathematical solutions to evaluate them. Monte Carlo methods, on the other hand, simulate the actual system processes and the random nature of the system behavior for evaluation.[Ref 1]

Existing Analysis Tools

Survivability analyses can currently be performed on ships and ship systems using tools such the Ship Survivability Model (SVM) and the Computer Aided Design of Survivable Distributed Systems (CASDiS) programs.[Ref 16-19] These applications are available at the Carderock Division of the Naval Surface Warfare Center (CDNSWC) and the Annapolis Detachment of CDNSWC (AD/CDNSWC).

The SVM evaluates ship survivability as a function of vulnerability and susceptibility using a comprehensive probabilistic assessment of the types of attack, the location of the hit, the damage from the hit, and the effect on the ships' systems including the integrity of the hull. It utilizes an extensive and often classified description of the ship construction, the types of attack, the effects of the various weapons, and the damage which can be expected for each type of hit. The output of this type of comprehensive analysis is assumed to be available for proper distribution system analysis using the methods described in this thesis. In the absence of this type of information, damage from hits is estimated using simple rules with the capability to add more extensive damage models to the thesis model by changing or replacing program modules.

The CASDiS program is a deterministic evaluation tool which also incorporates a detailed ship design specification for both the ship construction and the system logical and physical arrangement. It is intended to provide a method to allow survivability assessment of distributed systems at the detailed and contract design levels to meet Navy contractual specifications. The damage extent is input by the user and the resulting system alignment is evaluated automatically. Although damage and survivability are best analyzed as probabilistic events, the program has no probabilistic assessment capability since this is not necessary to meet Navy contract specification.

Program Development

The new methodology is coded as an enhancement to an existing reliability, maintainability, and survivability (RMA) analysis program called GATOR. GATOR was developed as a large system RMA analysis program for the U. S. Navy at the University of Florida in 1984. The new analysis package is named BEAVER[1]. The development of BEAVER is a direct response to the U.S. Navy's proposed new concept Advanced Electrical Distribution System (AEDS) to be implemented in future ship designs. The need for increased overall ship affordability, reliability, and survivability drive the requirements for the electrical distribution system changes.

The niche that BEAVER occupies is complementary to programs such as SVM and CASDiS. Acting as a powerful preprocessor, the BEAVER program is intended to provide a design tool which is used in the early stages of design at the conceptual level. It does not require the large overhead of ship construction detail specification, only a general specification of arrangement in relation to major ship survivability design aspects such as watertight bulkheads and decks. The program is designed as a desk top personal computer, or PC, compatible application to allow easy access for the system designer. Series of alternate distribution structures and control schemes can be quickly evaluated to allow trade offs to be made to aid in early system design selection decisions.

A primary innovation of BEAVER is its capability to perform *dynamic reconfiguration* with respect to either damage or normal operation. The program can not only assesses the damage impact on the system and all subsystems, but it can also reconfigure the system according to the control strategy input by the user (e.g., the user designates certain circuit breakers as controllable elements in the system). As a

[1] Since the GATOR was named after the University of Florida mascot, this program was named for the MIT mascot, the beaver.

consequence of this feature, the user can rapidly determine whether a proposed control strategy does, in fact, improve system survivability. Thus, series of alternate distribution structures can be quickly evaluated to allow trade-offs to be made in early system design selection decisions thereby driving the design towards a *balanced* configuration.

The ability to include probabilistic damage assessment is an enhancement which incorporates an important aspect of survivability design. Because the user may choose to create a damage scenario either deterministically or through Monte Carlo sampling, the evaluation results from BEAVER can be ported to either CASDiS or SVM type applications, respectively, in the final design stages when the system is required to meet explicit specifications. The enhancement contribution may not be fully realized, however, until a method can be determined to apply probabilistic evaluation techniques to the contractual specification. When detailed design is complete, the systems can then be analyzed with both SVM and CASDiS type applications to refine the system design to meet the specific application in the final ship design.

Chapter 2. Survivability Concepts

Survivability is defined as the capacity of a ship to absorb damage and maintain mission integrity. It is a function of *susceptibility* and *vulnerability*. Susceptibility is the degree to which a ship system is open to effective attack due to one or more inherent weaknesses. Susceptibility can be affected by changing such things as radiated noise, threat warning, or tactics. Vulnerability is the characteristic which causes a system to suffer degradation as a result of the effects of an attack. Vulnerability is affected by concepts such as component redundancy, location, and damage protection. The survivability analysis put forth in this thesis deals only with the investigation and evaluation of system vulnerability as it affects survivability.

Motivation

The United States Navy has long been aware of the importance of survivability in ship design and operation. The United States Congress enacted Public Law 95-485 on Navy Shipbuilding Policy due to the concern for the ability of combatant ships to withstand battle damage. U. S. Navy policy on survivability was put forth in a Chief of Naval Operations Instruction 9070 in response to this Public Law.[Ref 16] The major objectives of this instruction are to implement definitive policy which emphasizes the need for incorporating survivability features into the ship design, to establish minimum levels of ship survivability for use in ship design specifications, and to provide a basis for developing affordability and mission effectiveness measures for implementation throughout the ship design, procurement, and operational stages.

Combatant ships are expected to perform in battle, sustain damage, and survive. The total ship must be designed to withstand damage which can occur for many threat scenarios. The ability to effect major survivability improvements becomes more difficult once the basic design trade offs have been accomplished and the decisions have been made early in the design process. A "forward fit" strategy is important to achieve the best pay off in system design. Incorporation of survivability features early in the design stages can assist in ensuring an affordable balance of desired features in each new ship class design. System modularity in physical packaging and arrangement are aspects which are frequently analyzed for survivability. Just as important is the organization of the control architecture which is often left to the tying together of the distributed components after the system is designed and laid out. By direction, "survivability shall be considered a fundamental design requirement of no less significance than other ship characteristics such as weight and stability margins, maneuverability, structural integrity, and combat systems capability".[Ref 16]

A U. S. Navy Survivability Review Group (SRG), which met after the USS STARK incident, concluded that survivability can be improved with the incorporation of many considerations including redundancy and separation in the arrangement of distributed systems. Several of the findings apply directly to many existing distributed systems [Ref 16]:

- Provide redundancy and separation of selected vital spaces and vital components and equipment whenever ship size permits.

- Establish formal arrangement design standards for essential survivability features.

o Require horizontally and vertically separated and redundant cable paths where need is identified by deactivation diagram analysis.

o Utilize distributed processing, distributed systems, and local control to improve ship survivability.

o Use functional deactivation diagram analysis to ensure that survivability is designed into all combat systems for all new ship designs.

o Separate vital redundant auxiliary machinery that supports mission capability so that at least 50% of the capacity will survive a single weapon hit.

o Provide separated, redundant sources of supply and distributive systems, complete from source to user, for all hull, mechanical, and electrical (HM&E) support that is vital to the operation of the combat systems.

o Require survivability analyses be performed, against selected design threats, for all vital systems during all Naval Sea Systems Command (NAVSEA) conducted design phases.

o Require survivability analyses be performed as part of detail design for all vital systems.

As a result of the SRG findings, some of the survivability design features of the U. S. Navy destroyer, the USS ARLIEGH BURKE (DDG-51), were reviewed. Several improvements in the architecture of distributed systems were proposed including the layout of the electrical distribution system, the rearrangement of cable ways, and a reduction of the number of distribution cables. These improvements are specified to result in more affordable system with reduced vulnerability. The AEDS concept proposal is a result of this review. A review of conventional electric distribution system architecture is presented along with the major characteristics of the AEDS in the context of system survivability.

System Architectures

Although many shipboard systems, such as electrical distribution, fire protection, combat systems, and other support services, are distributed throughout the ship, the arrangement of conventional systems is usually based upon functional consideration, convenience, or system integration compromise. These systems must be designed to handle casualties not only due to equipment failure, but for fire, flooding, shock, or weapon hit as well. System protection must be balanced with the need for ship survival.

The scope of the problem includes not only reconfiguration of the electric plant, which can take from seconds to minutes, but the restarting of the loads which lost power. If the power was lost to the combat system, the components must be re-energized, the system must be reconfigured, and the detection, identification, tracking, and engagement of threat targets must be recommenced. Thus a casualty to the electrical distribution system can leave the ship and crew vulnerable to attack with no chance for an offensive capability.

It is this type of scenario that the electrical distribution system designers must consider when designing survivable distribution systems. This includes a change in the architecture of the system arrangement and incorporation of automated prioritized load shedding, system reconfiguration, and availability of alternate uninterruptible power sources. The incorporation of survivability concepts leads to a change in design concepts from conventional arrangements to the design of distributed systems, such as the AEDS, which can sustain damage and continue to provide the services necessary for mission continuation, albeit at reduced capability.

Conventional Systems

Circuit isolation and system reconfiguration are handled by a variety of means, some of which are invoked automatically and some manually. Isolation is most commonly handled by selective tripping of circuit breakers. Trip points are set to shed load to isolate the fault with the least effect to the overall system. Load shedding is a predetermined function of the way the system is configured for the desired initial reaction to the casualty. Vital loads are set to remain on line at the expense of non-vital loads. The electrical operator must manually reconfigure the system once the initial response is complete.

Reconfiguration can be accomplished automatically to some extent by the use of automatic bus transfer (ABT) switches. The ABTs are connected to one bus with an alternate power source, generally another generator supplied bus, available for transfer in the event of a casualty. Manual bus transfer (BT) switches are more commonly used. The electrical operator manually transfers power to the alternate bus as required. Reconfiguration can also be performed by placing an alternate generator on line to get power to the downed bus. This requires the operator to check for clearance of the faulted condition and to realign of the electric plant to redistribute loading.

A typical a..angement of a conventional electrical distribution system is depicted in figure 1. The system uses three 2500 KW gas turbine generators (GTG) as power sources. The system is a single zone with a radial architecture. The BT indicates a typical connection method for allowing transfer of power to alternate buses. There are 114 BT on the current DDG-51 arrangement. This provides two possible paths to each vital load. The system can be run in parallel or split plant as desired. Operational rules provide guidance on the plant configuration used for a given situation.

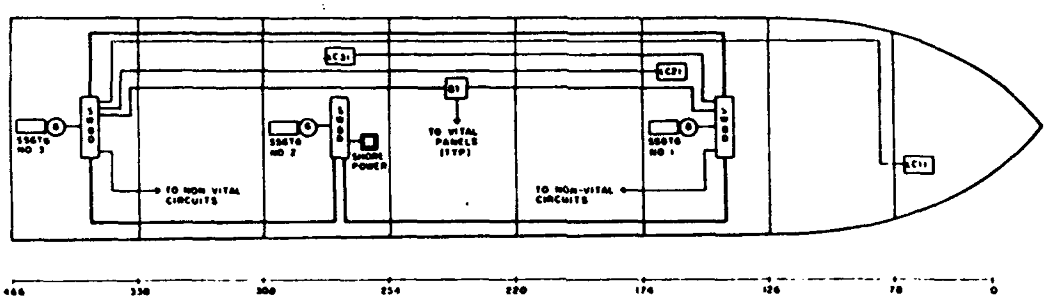

Figure 1. Existing DDG-51 Arrangement [Gibbs & Cox]

A typical cable routing scheme is pictured in figure 2 for a hull cross section. The main buses are horizontally separated port and starboard and vertically separated by two deck levels. Many other buses are run through other parts of the ship to supply load power. There are a total of 594 feeder cables supplying power to various loads.

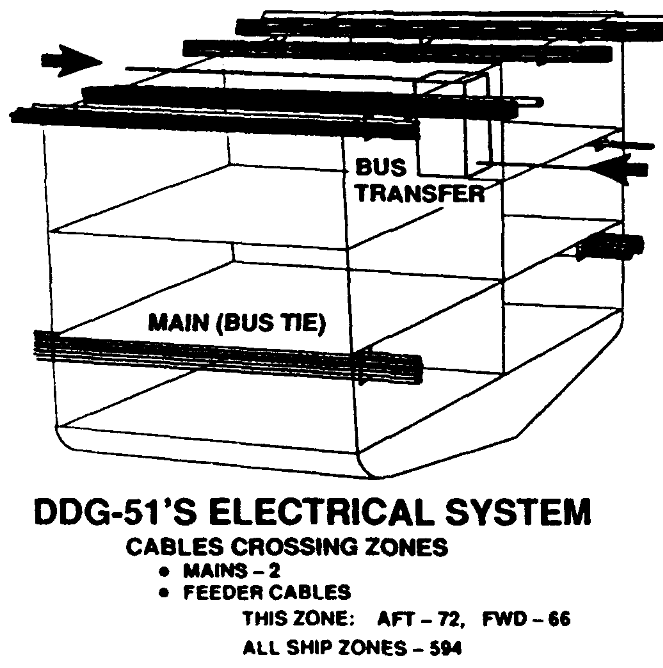

Figure 2. Typical DDG-51 Hull Cross Section [NAVSEA 05Z]

Survivability is addressed by the horizontal and vertical separation of main bus cables and the physical separation of the generators. This arrangement is vulnerable to damage due to the number of cables traversing the ship at many levels and the large number of BT switches which must be controlled manually to maintain power continuity. Practically any hit on the ship will produce some cable damage which will lead to loss of loads.

Advanced Concept Systems

The U. S. Navy's near term[2] AEDS concept consists of redesigning the electric power system arrangement including cable layout, zonal distribution, and control system architectures. The far term concept proposes the application of an automated control capability. These new system configuration concepts need to be analyzed to quantify the reliability and survivability costs and benefits. Affordability can be quantified as a measure of both acquisition and combat effectiveness costs of alternative designs. This includes the effects of system zone division, power source and distribution equipment arrangement, and communication, command, and control (C^3) data bus system architecture design including the effects of the implementation of automated control.

Rearranging the system improves affordability in several areas. The layout scheme can reduce the number and length of cable runs which reduces the initial acquisition cost. The cable reduction also saves weight which allows ship size and power requirements to be reduced. The ship size and power reduction improves both acquisition and life cycle costs.

[2] Near term development is within 5 to 10 years with far term extending beyond.

The incorporation of a zone concept improves ship producibility which reduces shipbuilder basic construction cost, thus reducing acquisition cost. Producibility is enhanced since cables can be installed within the production zones as they are outfitted. As the ship is assembled from the zone modules, the majority of the cables are then connected between modules rather than strung through the ship after the hull is completely erected.

The proposed arrangement, along with the new control system architecture and the implementation of automated control, provides increased electrical power system survivability during casualty events. Distributing the electric system power generation components throughout the ship provides a physical separation which prevents single hits from knocking out the entire system. Providing a control hierarchy which is capable of system control at a zone level as well as the whole ship level supports this physical arrangement. Reducing the number and length of power distribution cables reduces the probability of distribution system interruption due to cable damage. Automation of control allows for improved power continuity during casualty events.

The AEDS is based on the concept of a distributed architecture which consists of arranging systems considering separation and redundancy. The use of ABT switches rather than manual BT switches improves casualty response to power outages. The operation of the electrical distribution system remains virtually the same with manual paralleling of generators and switching of major bus ties. The redistribution of the power by arrangement of cables and load centers (LC) improves overall system survivability by physical arrangement alone. Far term architectures include provisions for automated paralleling of generators, automated switching of bus tie breakers, and the use of uninterruptible power sources (UPS) to supply vital loads during casualty operations. The UPS is implemented as a battery energy storage system (BESS) which supplies interim power for a short time on the order of 10 minutes while the electric distribution system is being reconfigured to allow normal powering of the vital loads.

Figure 3 is a simplified diagram of the conceptual electrical distribution system. The system is characterized by the zonal distribution with zones separated by watertight bulkheads. Each zone has only 4 penetrations for main bus power. Each zone also has 2 load centers, each fed from 2 sources. This creates a doubly redundant source of power to each zone. The ABT switches to the vital loads reconfigure power automatically to maintain power continuity. Manual BT switches can be used whenever the automated function can be deemed unnecessary. The system can be operated with all sources in parallel or in split plant depending on the mode desired.

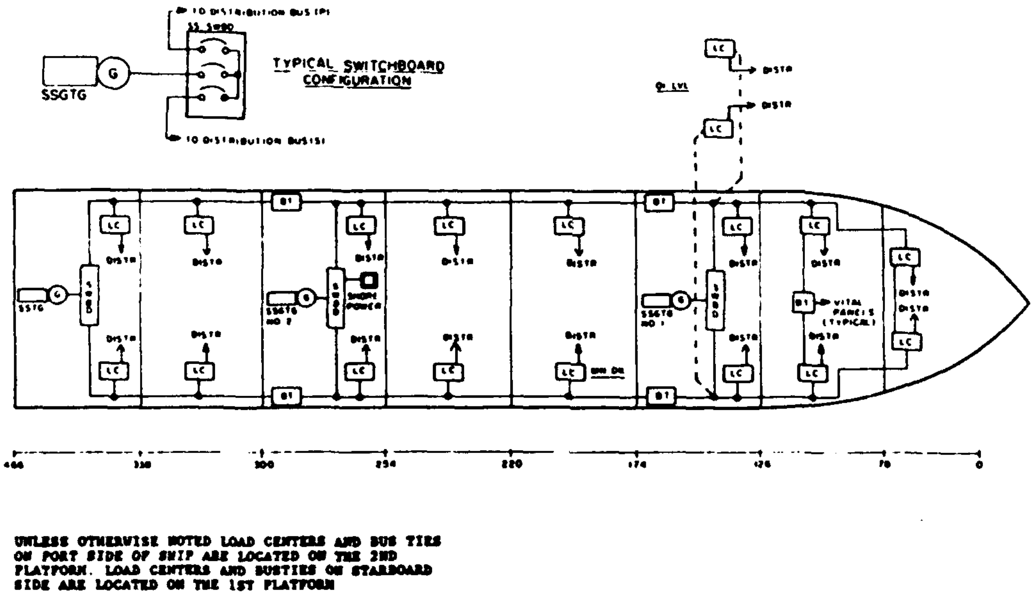

Figure 3. Conceptual Zonal Electrical Power System [Gibbs & Cox]

The arrangement of the cable penetrations and the load centers is shown on figure 4. Physical separation of cables and load centers is maintained, however, there are no feeder cables penetrating the watertight bulkheads. Components within the zone are all powered from load centers within the same zone. The drastic reduction in the number of cables which are run between watertight compartments greatly reduces the chance of a casualty in one zone affecting the power flow to the other zones. Damage effects can be contained in individual zones rather than affecting systems outside watertight zone boundaries.

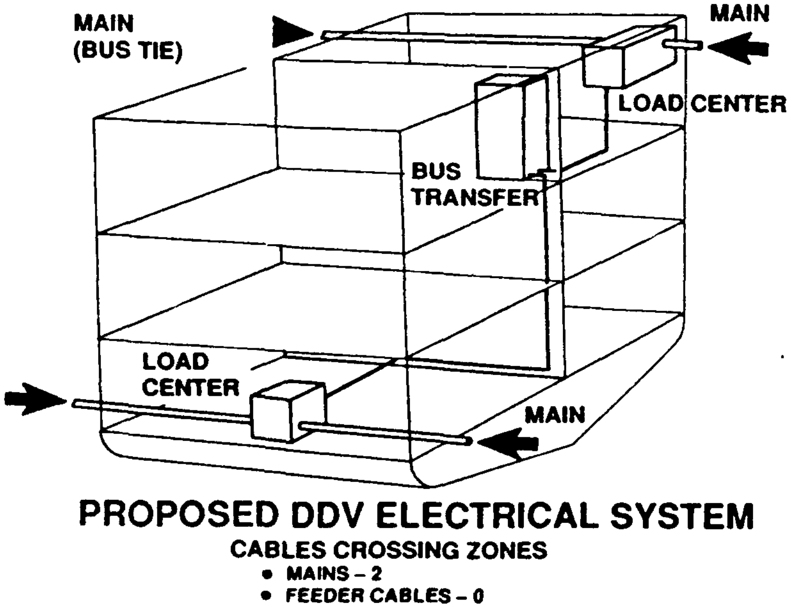

Figure 4. Conceptual Zonal Hull Cross Section [NAVSEA 05Z]

Zonal load level and location of watertight bulkheads is depicted in figure 5 . The loading indicates the maximum KW load in each zone. The zones are separated by watertight bulkheads.

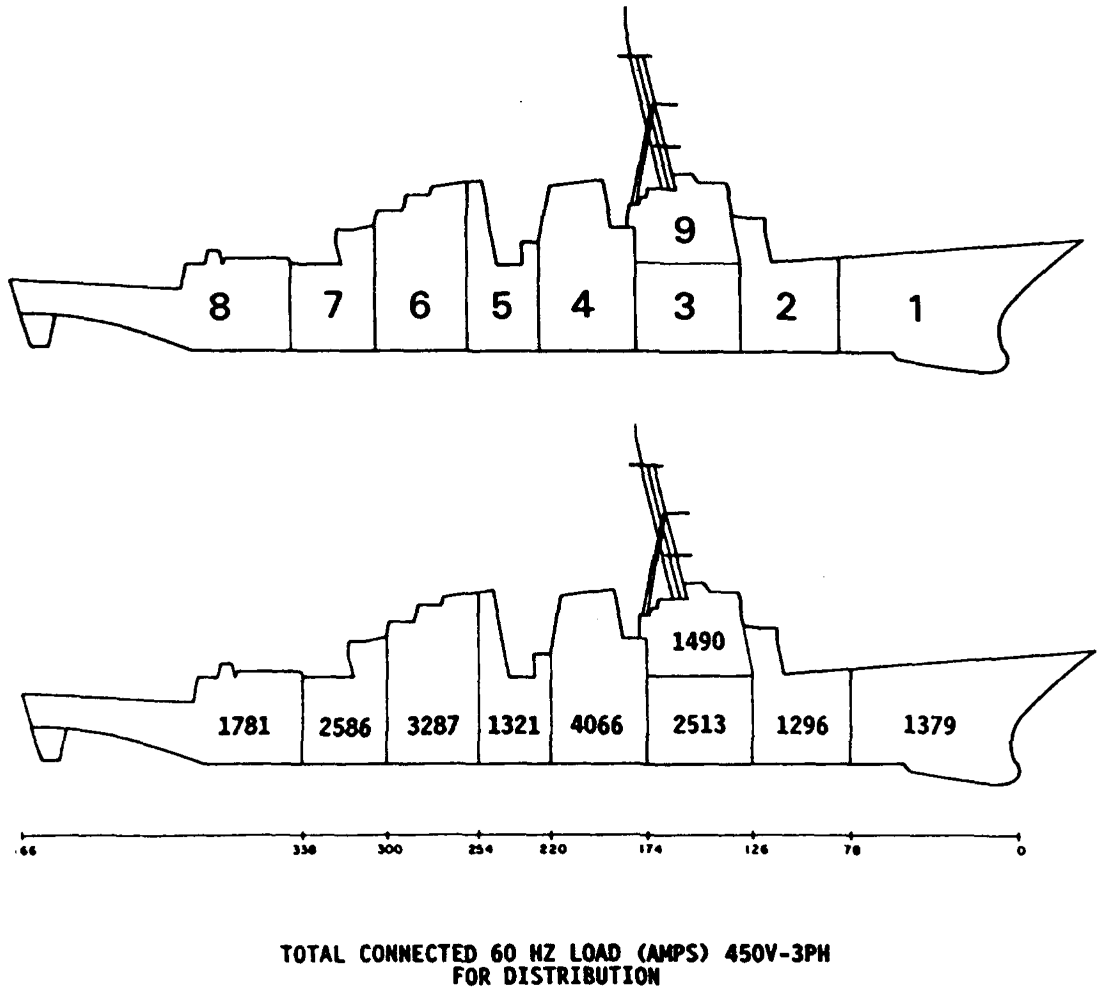

Figure 5. Zone Division and Load Distribution Diagram

The proposed distribution system implementation appears to be more survivable than the conventional system based on the arrangement differences, but is it the best arrangement? What are the key features which make this zone concept more survivable? How do control strategy differences affect survivability? Are there ways to refine the solution to further improve the system? What is the most cost effective way to provide improved survivability? These and other questions can be answered with the aid of an analysis tool which can measure the survivability of alternative distribution system arrangements.

Chapter 3. BEAVER Methodology Development

The program development is concerned with applying a probabilistic method to evaluate distribution system survivability in the event of a casualty to the ship. The evaluation consists of determining a survivability index for a given system design given a damage distribution. The analysis is an adequacy assessment to determine if there are enough assets to supply power and if there are continuous transmission paths to supply power to the loads. The evaluation does not analyze the stability of the system during transient states, but assumes the system can reach a stable steady state condition without the transients causing additional detrimental system effects.

The affordability issue is addressed by considering the system acquisition costs, combat effectiveness costs, and survivability indices resulting from the analyses of alternate system designs. Acquisition cost is an estimate of the component procurement and installation costs. Combat effectiveness cost is an estimated index associated with the cost of load outages on the overall combat capability of the ship. Load outages are measured from the system alignment resulting from the known casualty events. The outages can be grouped into areas such as fire control, weapons, life support, propulsion, or any other ship systems which can be logically grouped together. Combining these costs with the survivability indices of the alternate system designs provides a quantitative measure of the incremental cost of providing improved survivability. RMA affordability determinations are made in the same way.

The current survivability assessment tools, the SVM and CASDiS programs, use extensive system descriptions to analyze the survivability of a particular system arrangement. These methods are useful in the final stages of system design when the ship and associated systems are well defined and details are generally known. These programs are also preferred for conversion of existing system design since the ship being converted generally has the detailed specifications available. BEAVER has been developed for relatively convenient system design trade offs at the conceptual level. The new program is a modification to an existing RMA evaluation program originally named GATOR.

GATOR was used by the Naval Underwater Systems Center (NUSC) to evaluate the BSY-1 submarine combat system RMA. The program was modified in 1988, and renamed DISE, by the Charles Stark Draper Laboratory (CSDL) to handle fault tolerant system RMA. DISE has been recently used to evaluate some of the CSDL unmanned underwater vehicle (UUV) systems and some National Aeronautics and Space Administration (NASA) space station systems. The BEAVER program modification to GATOR, the new subroutines of which are listed in Appendix C, includes survivability analysis using either static or dynamic system reconfiguration capabilities.

The original GATOR program was designed as a FORTRAN coded simulation tool to analyze large, complex systems for RMA attributes. It has additional capabilities to perform phased missions, priority classification of repairs, and general resource sharing. Although originally designed for use on large computer systems, it has successfully been converted to PC system use. The size of the system to be analyzed is limited only by the ability of the memory of the computer system to handle the array dimensions. The program employs a general fault tree structure and a Monte Carlo method to simulate the random nature of failure events.[Ref 20]

Development Philosophy

The guiding philosophy in the development of BEAVER was to keep as much of the underlying code and structure of GATOR and DISE intact, to complete the survivability and reconfiguration routines in a modular form for use in either DISE or GATOR, and to utilize as much of the original input file structure as possible. Both the GATOR and DISE programs have proved that the methodologies employed are sound and well tested. Using the same structure in the BEAVER implementation saves verification time for some of the new analysis techniques. Using existing structure saves a great deal of effort as well by not reinventing the wheel, since the GATOR RMA analysis methodology is quite well suited to the survivability analysis methodology required. The combined package of RMA and survivability analyses also offers a convenient environment to evaluate systems for multiple attributes with a single input file. Thus the BEAVER program logic and methodology remains the same as the GATOR program. The conversion of the code to C or Pascal language, which is well suited to tree structuring and linked lists, was not considered to maintain compatability with existing code.

Methodology Employed

The methodology uses a fault tree structure as a basis for evaluation. The system to be analyzed is represented as a logical construction of subsystems which are connected in series or parallel and organized to represent actual system functional connectivity. As components fail, the effect on the entire system is evaluated. Statistics are compiled over multiple system operations, or missions, on either an event driven time scale basis for RMA analysis or a single instantaneous event basis for survivability analysis.

The reconfiguration control portions of the program are meant to provide various information and control capability to the system designer. The development of a structure which can be used in many different scenarios is considered more important than implementing and testing specific control rules. The actual control strategies incorporated into the system design are to be determined by the user by adjusting the source code to access the system status and control features provided by BEAVER. The source code programming should only involve the rules which determine the handling of system faults and not the ability of the program to detect, isolate, and reconfigure components.

Monte Carlo Usage

Monte Carlo methods are employed for the RMA analysis routines to simulate the non deterministic nature of equipment failure. The existing RMA analysis is event driven. The program determines if the next event is a component failure or repair. A component is selected randomly based on it's relative probability of having a transition as a function of overall system rate. The component is then processed as either a failure or repair.

The survivability analysis program has the ability to use a Monte Carlo analysis method to simulate the probabilistic nature of a damage event. Monte Carlo analysis is used in BEAVER since the GATOR program, used as the basis for BEAVER development, incorporates a fault tree structure and a the Monte Carlo method. The distribution of the damage profile is determined by the user since damage events can be described in many different distributions for many different scenarios. Monte Carlo methods are not required for survivability analysis to allow BEAVER to be employed as an evaluation tool useful in determining survivability in a U. S. Navy contractual deterministic sense. In the deterministic case, the region of damage is defined by the user by indicating downed equipment in the input file.

Fault Tree Structure

The fault tree structure is built up from individual pieces of equipment. The individual components are grouped logically to form subgroups as series or parallel combinations. The subgroups are then grouped into a complete fault tree with the individual pieces of equipment as leaves at the bottom of the branches leading up to the final system combination at the top. The concept is similar to the building of the system deactivation diagrams used in current survivability analyses.

When a particular component transitions, either from up to down, for the damage or load shed type events, or from down to up, for putting components back into system use, the effect of the failure is traced through it's subgroup through to the top system node to determine if the system is operational. In the case of survivability analysis, the system status is noted after each damage event. The RMA analyses use multiple missions, each for a predetermined length of time, to estimate a value for individual component, subgroup, and overall system reliability and availability as a percent of the time spent in failure or in operation.

Control Strategies

The control implementation capability is important in determining overall system performance. For example, a control strategy could be defined as follows. After a casualty event is processed, check the powering of all ship loads. If sufficient power is not available to supply all loads within all disjoint regions, then determine a system alignment, using a predetermined set of rules, which can provide adequate power. In the event that no system alignment can be found which provides for adequate powering of all loads, perform load shedding by priority to attempt to achieve an alignment which provides power for all vital loads. Any event which results in a system alignment which provides sufficient powering of all defined loads within normal system operating parameters would then be considered as survivable.

For an investigation of automatic control implementation, the control system could be assumed to function to manipulate all of the system controls, including generator paralleling, as necessary to create an alignment which provides adequate powering. The real considerations associated with the actual automated system reconfiguration need not be addressed in this portion of the analysis. The reconfiguration processes required to achieve a *survivable system* alignment can be tracked to provide input to a separate analysis of what a control system would need to implement, such as automated paralleling of generators, to perform the transitions automatically.

Damage Events

The survivability analysis allows for multiple, simultaneous equipment failures. Each mission is represented as a single damage event. The algorithm differs from the RMA type of analysis in that the system status is not recorded until all damaged equipment effects have been evaluated. Survivability is evaluated over many single damage events with the determination made as to the percent of trials which the system survived.

The determination of which equipment has failed due to damage is made based on the region of the damage extent and a comparison of blast interaction strength to the level of shielding provided for the equipment. The damage interaction can be a function of weapon type, warhead size, hit location, component protection, and many other factors. Since a damage event can be described using one of many different damage distributions depending on the type of interaction , the BEAVER program provides a modular routine which can be modified to more accurately describe various damage events. The current distribution is modeled as a simple spherical region of interaction.

The blast magnitude is modeled as an exponentially decaying function, as shown in figure 6. Actual blast functions vary widely with many different variables, so this representation is simply meant to portray a unitless "magnitude" of the blast and the decay of the magnitude with distance from the blast center. The actual functional form of the blast can be modeled by real distributions if that information is available. The parameters which characterize this exponential can be modified in the input file as desired.

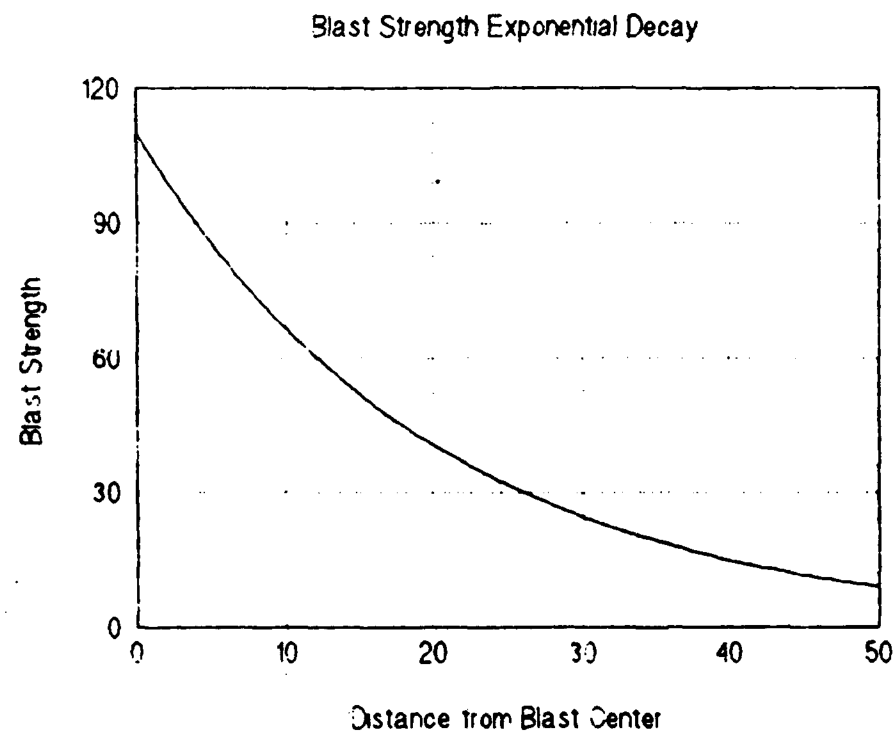

Figure 6. Blast Magnitude as a Function of Distance from the Blast Center.

Survivability Determination

Survivable events are defined by the user by the logical arrangement of the system. The survivability determination is made by determining if the system has sufficient power capability to supply connected loads within desired operating parameters. The up and down status of components, subgroups, and the system node are kept by the system. The number of survivable trials is converted to a percent survivability by dividing by the total number of trials to provide the survivability index.

The survivability of individual components can be used to indicate particularly culpable individuals. By logically arranging subgroups into the same organization as the ship systems, the survivability of each functional subsystem, such as fire control or propulsion for example, can be tracked. Tracking these subgroups can lead to determination of combat effectiveness indices from the percent survivability of the subgroups.

Affordability

The affordability of a system is measured by the acquisition cost of the system components. Dividing the survivability and combat effectiveness indices by the total system or subsystem cost can be used as an indicator of the incremental cost of providing improved survivability and combat effectiveness. Although this is not a complete cost analysis, it is a useful indicator which can be used to make a relative comparison of alternative systems.

Chapter 4. BEAVER Program Operation

The BEAVER program modification to GATOR includes survivability analysis and dynamic system reconfiguration capabilities. The combined package of RMA and survivability analyses offers a convenient environment to evaluate systems for multiple attributes with a single input file. Figure 7 illustrates the complete BEAVER/GATOR combination program. The program operation is summarized in the **Program Overview** section with specific subroutine descriptions and data structure details in the proceeding sections.

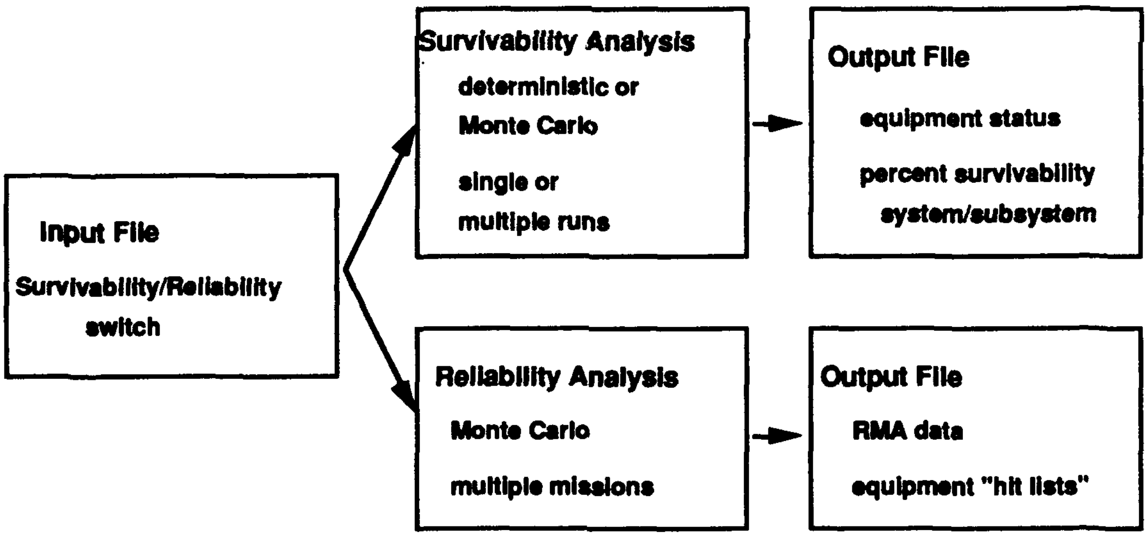

Figure 7. The BEAVER Program

Program Overview

Figure 8 illustrates the basic BEAVER flowchart . Four major new subroutines, LINEUP, DAMAGE, POWERPATH, and CHKLOAD, have been added to the GATOR program, and the original MAIN routine has been substantially modified to allow the user to exercise the various options. Notice that the user can deterministically provide the damage profile through the subroutine LINEUP or can use Monte Carlo sampling in the DAMAGE subroutine. All functions are selected for use in the input data file.

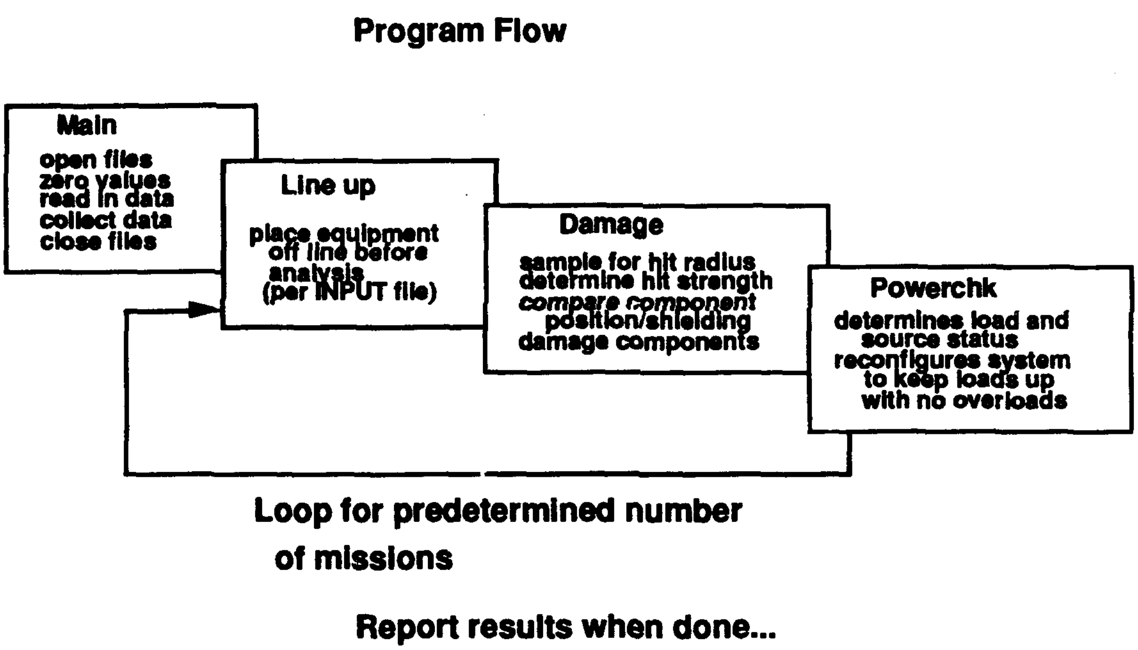

Figure 8. Basic BEAVER Flow Chart.

Equipment data such as component mean time between failure (MTBF), mean time to repair (MTTR), repair priority, cost estimates, component blast protection hardness, source or load power level, and location is arranged in the input data file. The program reads the input data file and puts together the tree structure. For survivability analysis, the hit location and damage extent parameters along with the equipment locations must be specified. All points are referenced to the forward perpendicular (x = 0), the centerline (y = 0), and the ship baseline (z = 0). All components except for cables are input as single (x,y,z) points. Cables have two points, an origin and a termination. The cable is then considered as a straight line between the two points. It is not necessary for program operation for the components and the cable end points to terminate at coincident points.

The LINEUP routine takes the initial system lineup and sets up the equipment for program runs. A new routine named CSTAT, for control status, is called by LINEUP each time a piece of equipment is switched off. The CSTAT routine is necessary to allow equipment to be switched off without it being considered a failure or damage event. A note listing each piece of equipment which has been switched off before the survivability analysis begins is sent to the output file. After LINEUP, the program then invokes POWERPATH which traces the beginning tree structure to provide an array of all the sequential power paths from the loads to the sources. This array is important in testing system power path continuity and isolating system faults since the tree structure does not provide this information directly.

The program then determines the extent of the damage. The survivability analysis required development of a new algorithm to allow for multiple, simultaneous equipment failures. This is coded in DAMAGE. Each mission is represented as a single damage event. The determination of which equipment has failed due to damage is made based on the region of the damage extent. If a Monte Carlo simulation is used, the damage extent is calculated as a given volume for each particular run. The distribution of the damage extent can be determined by the user in the formation of the damage distribution profile programmed into the application. The current algorithm produces a set of damage radii creating spherical volumes uniformly distributed between a minimum and maximum radius as input by the user. More physically descriptive damage profiles can be programmed if data is available.

Any equipment within the damage region, which has a damage interaction that exceeds the hardness level of the component, is marked as failed and the tree structure is evaluated for the net effect on the entire system. This is a straight forward determination for all point located pieces, but the cables are stored as straight line segments. If any portion of the line is within the damage region, the entire cable is failed. The cable radius to blast center location calculation is performed by finding the distance from the hit point to the cable line. If the distance is less than the damage radius and the blast magnitude at the interaction radius is greater than the cable shielding value, the cable is failed. All possible cable orientations are checked to ensure the proper distance to the line segment is found.

When Monte Carlo sampling is used, survivability is evaluated over many single damage events with the determination made as to the percent of trials in which the system survived. The program loops through a prescribed number of trials, resetting the system after each hit. Statistics are kept after each trial as to the status of equipment, subgroups, and the system node as to whether they are up or down. If the system remains up, the trial is considered survivable. The total number of survivable trials is divided by the total number of trials to determine a percent survivability.

For the deterministic case, the damage extent is fully described before evaluation by specifying which components are damaged by switching them off in the input file. The program then runs through one evaluation and lists which components are up or down and whether the system and any indicated subsystems survived overall. A feel for the survivability of the system can be obtained by moving the damage region around and recording the result.

Once the system has been damaged and the effect has been accounted for in the fault tree evaluation, the POWERCHK routine is called. POWERCHK handles the evaluation of the power system. It invokes CHKLOAD to determine which loads are powered, which generators are paralelled, and the loading on each generator. POWERCHK exists as an example of where a control algorithm would be placed. The code currently contains only a test for the powering status of the loads and the loading level of the generators. The load warnings indicate whether or not loads should be powered or that they cannot be powered due to system damage. The power source warnings indicate which generators are overloaded.

The program data is converted to output information in the RESURV subroutine. This section calculates the percent survivability statistics for each component and subsystem. The set of subsystems to be tracked can be identified in the input file. All components and subsystems have data tabulated as the default setting.

Subroutine Descriptions

The set of subroutines developed for BEAVER are presented in detail. These routines splice into the DISE program at CSDL, as well as GATOR. The differences are minor and the structure used by GATOR is maintained. The new versions of the MAIN and the READIT routines should be used in place of the GATOR versions to ensure compatibility. Any programmers familiar with the GATOR algorithms should have no problem combining the BEAVER and GATOR programs.

MAIN Routine

The Main routine controls the program looping and administrative functions. Calls to open and close files, initialize variables and arrays, read in data, process events, generate reports, and close files are performed. The data file is read in and the radius from the hit center to each of the components is calculated as the simple distance between two points in space.

The distance from the hit location to a cable is handled differently. Cables are modeled as line segments and are input by noting both end points. RCABLE calculates the closest point on the line segment to the hit location so damage determinations can be made correctly.

The program then flows into a loop which is performed once for each mission as specified in the input file. INITAL is called to initialize the system for evaluation. LINEUP is called after INITAL to allow for equipment to be turned off before analysis begins. The setting of the survivability switch "SURVIVE" in the input file causes the loop to skip over the RMA analysis routines to the DAMAGE routine. This is where the system is damaged and the effect of the damage is determined for the rest of the system. POWERCHK is then invoked and the assessment and realignment of the system is performed as specified in the control strategy.

The single phase of the survivability assessment causes the PCHANG routine to terminate the mission. During RMA analysis, each event must be processed and the time tested for potential phase changes. Survivability determination is made using only one event per mission. The looping is done over a number of missions to determine the survivability indices.

At the end of each mission, the statistics are closed out. The components which were damaged are noted as failed components. When the total number of missions has been completed, the analysis run is complete and the REPSURV reports the survivability of the system and the incremental cost of survivability. The output is made to a file named "BEAVER TALE". If consecutive runs of the program are to be made, it is suggested that the output files be renamed before they are overwritten during subsequent runs.

The program is currently configured to analyze one hit location point at a time. A looping program control arrangement to distribute damage impact points in a realistic distribution around the ship structure would provide a more complete survivability assessment. At this point, the program operation is proved as a single point hit algorithm.

RADIUS (number of parts) - keeps a list of the hit center to component radius for each piece of equipment. The value for cables is found in the RCABLE subroutine.

RCABLE

Cables are input as straight line segments. Since the minimum distance from the blast center to the cable will be the damage interaction point, the cable to blast center must be calculated using a geometrical distance in three dimensional space from a point to a line segment. RCABLE finds this distance and assigns it to the cable. CX, CY and CZ together with the 1 or 2 index are the end points of the cable. HITX, HITY, and HITZ are the blast center locations.

```
            AA = CX2 - CX1
            BB = CY2 - CY1
            CC = CZ2 - CZ1
C
            TTT = AA*(HITX-CX1)+BB*(HITY-CY1)+CC*(HITZ-CZ1)
            TT = TTT/(AA**2+BB**2+CC**2)
C
            XX = AA * TT + CX1
            YY = BB * TT + CY1
            ZZ = CC * TT + CZ1
C
13050       IF ( XX .GE. CX1 ) .AND. ( XX .LE. CX2 ) GOTO 13100
            GOTO 13300
13100       IF ( YY .GE. CY1 ) .AND. ( YY .LE. CY2 ) GOTO 13200
            GOTO 13300
13200       IF ( ZZ .GE. CZ1 ) .AND. ( ZZ .LE. CZ2 ) GOTO 13400

13300       CXX = CX1
            CYY = CY1
            CZZ = CZ1
            RADSQ = (HITX-CXX)**2+(HITY-CYY)**2+(HITZ-CZZ)**2
            RAD1 = SQRT(RADSQ)
            CXX = CX2
            CYY = CY2
            CZZ = CZ2
```

```
            RADSQ =  (HITX-CXX)**2+(HITY-CYY)**2+(HITZ-CZZ)**2
            RAD2 = SQRT(RADSQ)
            IF ( RAD2 .GT. RAD1 ) THEN
                 RADIUS ( I ) = RAD1
            ELSE
                 RADIUS ( I ) = RAD2
            ENDIF
            GOTO 13500
C
13400       CXX = XX
            CYY = YY
            CZZ = ZZ
            RADSQ =  (HITX-CXX)**2+(HITY-CYY)**2+(HITZ-CZZ)**2
            RAD = SQRT(RADSQ)
            RADIUS ( I ) = RAD
```

READIT

Subroutine READIT simply reads in the data from the input file, counts certain pieces of types of equipment, and sums up system cost. There are several arrays built in this subroutine which are of general importance.

EQTYPE(number of parts) - This array contains the BEAVER type code for each piece of equipment. Note that this is the new BEAVER implementation of equipment type which is different from the GATOR equipment type. EQTYPE is used as a flag to differentiate types of system components for handling as necessary, such as for retrieving the second data point for a cable or for finding controllable components in the power path.

EQDATA(number of parts) - This array holds all the information in fields 3 through 12 of the input file equipment data matrix. Chapter 5 has a detailed description of the EQDATA fields.

HITDATA(6) - Contains the hit location, strength, distribution parameters, and decay constant. Chapter 5 has a more detailed explanation of the information in this array.

SOURCE (number of sources), **LOAD** (number of loads), **BREAKER** (number of breakers), **CABLE** (number of cables), **CABINET** (number of load centers), **BUSTIE** (number of bus tie breakers), **ABT** (number of automatic bus transfer switches), **ALTSRC** (number of alternative or backup emergency power sources) - These arrays contain the equipment number, EQUIPNO, assigned to each of the members of the array. The EQUIPNO is assigned in the input file. The arrays provide a way to find any of the given components in the tree structure.

LINEUP

The function of subroutine LINEUP is to initialize the system to the line up described in the input file. This includes checking the initial status flag in the EQDATA array and setting the UP flag to 1 to indicate that the equipment is off as directed by the input file. CSTAT is called to perform the actual setting of the UP flag. The system rate, for RMA analysis, is also adjusted so this routine could be useful in an RMA implementation of the new code capabilities. The rate is not used by the survivability analysis.

CSTAT

CSTAT is an important rewrite of the original GATOR STAT code. Its function is to change the state of a component from on to off (UP flag from 0 to 1) or from off to on (UP flag from 1 to 0). It then evaluates the effect on the rest of the system by tracing through the tree and putting down any subgroups fed from the component which had its state changed.

The major difference from the STAT routine is that the CSTAT routine does not adjust the system rate, nor does it keep track of the component and subgroup failure statistics since the on off transitions are not failures, but intended state changes.

DAMAGE

DAMAGE finds the blast damage region and determines whether or not the component survives the interaction. Calls are made to subroutines DFAIL and STAT and to the BAKST routines to place the component out of commission and check the effect on the status of the rest of the system. All components with radius less than the blast radius are tested for potential damage.

The main function is to determine the damage extent and decide which equipment has failed. The Monte Carlo determination of the damage radius is found by calling a random number generator subroutine RAND. This number is multiplied times the difference in the maximum and minimum radius of the blast as input by the user. The fractional distance is then added to the minimum radius to obtain the damage radius. The individual pieces of equipment are then tested to find if they are within the blast damage region. If they are beyond the damage region, they are not processed for any type of damage event.

Components which are within the damage radius are compared to the blast magnitude at the radius from the blast center. The SHIELD value input for the component, I, in the EQDATA array is compared to the HTMAG level at the component RADIUS(I)

```
SHIELD = EQDATA( I, 7)

HTMAG = HTMAG0 * EXP((-1) * DF * RADIUS(I) )
```

HTMAG0 is the hit blast magnitude at zero radius with DF being the exponential decay factor. Both of these values are input in the EQDATA array. If the hit blast magnitude HTMAG is greater then the shielding level for the component, SHIELD, then the component is failed. Otherwise it is left intact.

HITDAM (number of parts) - keeps track of which components were directly damaged by the hit. This helps determine if components were damaged, failed, or simply switched off by the user or a control algorithm.

POWERPATH

This routine traverses the tree structure and builds a sequential ordering of the tree elements to allow for isolating system power continuity disruptions. The data in the tree structure must be input as described in chapter 5 to ensure correct component sequencing. The output is the PWRPATH array which can be accessed to find downed components in the path from the load to source or source to load. This search through this array while checking the UP status, HITDAM status, and the desired control of breakers can help determine if breaker control will reapply power or not. All ABT loads show two power paths for each source since they are connected to allow switching to an alternate bus. Isolation of faults to a specific downed component can also be determined by examining this array. Note that the array is built showing all potential power continuity paths found by searching the entire tree whether components are on or off. The status of each component must be checked to determine if continuity exists in the current system configuration. The array SRCPATH is the column reverse ordering of PWRPATH to facilitate checking of source to load continuity for investigation of generator loading.

PWRPATH (number of paths found, number of parts) - this array holds the sequential component path from the loads to the sources. A zero entry terminates the list. The array is organized with the load number in the first column and the power source at the end of the string in the right hand column just before the last zero in the row.

SRCPATH (number of paths found, number of parts) - this array holds the sequential component path from the sources to the loads. A zero entry terminates the list. The array is organized with the source number in the first column and the load at the end of the string in the right hand column just before the last zero in the row.

CHKLOAD

This subroutine traverses the tree only along paths which have no downed pieces of equipment or subgroups along the way from the load to the source. All loads are checked as input from the LOAD array. An array, GENLOAD, is built to keep track of which loads are powered. The row index is the SOURCE number and the column index is the LOAD number. A one is placed in the matrix if a load is found to be powered from a particular source. A zero entry means no source, i, was found for that load, j. By summing along columns, the user can determine which loads are not powered due to the zero sum total. The LOADWN array keeps track of unpowered loads for processing by the control system

In this implementation of CHKLOAD, the PLLCHK array keeps track of the column sums. A one in PLLCHK(I) for LOAD(I) indicates that the load, I, is powered by only one source A number greater than one indicates the multiplicity of the sources supplying that load. This indicates that generators are lined up in parallel to this load, so the overall loading of the paralleled generators is adjusted to reflect the shared load. The generator loading is kept in the SRCLD array. The overload level of any generators is kept in the OVRLD array.

GENLOAD (number of sources, number of loads) - keeps track of load to source connectivity for determination of which loads are not powered and which generators are paralleled. Entries are 1's and 0's only to indicate that a load is powered from that source or not.

SRCLD (number of sources) - contains the load level of the sources.

PLLCHK (number of loads) - used as the column sum indicator of the GENLOAD matrix to determine paralleled generators and to calculate shared load.

LOADWN (number of loads) - keeps track of the loads are down and should be up as indicated by system connectivity and desired on/off status..

OVRLD (number of sources) - holds the amount of over load on the source, if any.

REPSURV

This routine simply gathers the data from the survivability runs and outputs the results to the screen and the output file, BEAVER TALE. Suvivability indices are calculated as a percent of survivable trials divided by the total number of trials. These indices are then divided by the total system cost to allow comparison with other trial results for alternative systems to find an incremental cost of providing a certain relative level of survivability. The subsystem to be tracked for output can be specified in the input file. If no set of equipment and subgroups is specified, then the entire system data is output.

Control Implementation Example: POWERCHK

This routine is where the system reconfiguration decisions are made. This routine is coded as an example of a potential control strategy. Other strategies would have to be coded in its place using the arrays and subroutines provided. The example is described to show a possible use of BEAVER in evaluating reconfigurable systems for survivability.

In POWERCHK, CHKLOAD is invoked to find the loads which are not powered, the generators which are paralleled, and the loading on the generators. Once CHKLOAD returns with this information, POWERCHK attempts to put all loads which are not currently on line, back on line by checking POWERPATH to find which lines are potential alternate paths. If no path is found, a message is output as to why the load could not be powered. If an alternate path is found, the controllable component's state is changed to allow powering. The routine must loop back to invoke CSTAT for each controlled component, one at a time, so that the effect on the rest of the system can be recorded. CHKLOAD must then again be invoked to determine the result on the load powering and source loading. In this implementation, paralleling of generators is not allowed since no automated paralleling is being evaluated. To test the effect of automated paralleling, the routine would just have to be allowed to control the tie breakers between the generators. The pseudo code which follows implements the control of ABT protected loads to ensure these vital loads are powered before any load shed decisions have to take place.

```
/*  FIND THE LOAD AND SOURCE STATUS  */
10  CALL CHECKLOAD
        DO  20 I = 1, NUMBER OF LOADS
/*  THE LOAD IS OK IF LOADWN(I) .EQ. 0  */
            IF <LOADWN(I)  .EQ. 0> GOTO 20
            ELSE IF <POWERPATH CHECK FROM LOAD(LOADWN(I)) TO
                ANY SOURCE(J) INDICATES AN ABT IS UP> .AND.
                <ALL OTHER COMPONENTS IN THE POWERPATH ARE UP>

/*  SHUT BREAKER EQDATA(ABT,8 .OR. 9) FOUND FROM
    POWERPATH CHECK.  NOTE THAT THE ACTUAL EQDATA FIELD
    BEING USED AS THE ARRAY INDEX HERE IS A REAL NUMBER  */

            THEN <CALL CSTAT( EQDATA(ABT, 8 .OR. 9), 0 )

                  CALL SYS_STATS

/*  GO BACK TO CHKLOAD TO RE EVALUATE THE SYSTEM LOADING.
    THE LOAD(LOADWN(I)) SHOULD NOT BE IN LOADWN(I) SINCE
    IT WAS CHECKED FOR POWER PATH CONTINUITY  */

                  GOTO 10>

            ELSE <OUTPUT WHY THE LOAD(LOADWN(I)) IS DOWN
                  WHETHER IT IS SWITCHED OFF, DAMAGED, OR
                  HAS A PATH DISCONTINUITY>
     20 CONTINUE
/*  ALL LOADS AT THIS POINT SHOULD EITHER BE POWERED OR THE
    REASON FOR THE OUTAGE SHOULD BE KNOWN.  NOW CHECK FOR
    GENERATOR OVERLOADING AND POTENTIAL LOAD SHED NECESSITY.
    AT LEAST ALL LOADS ARE ACCOUNTED FOR SO LOAD SHED CAN
    TAKE THEM ALL INTO CONSIDERATION BY PRIORITY IN
    SHEDDING LOAD.  */
```

Once all of the loads have been checked, the generator loading is checked. The loads are checked before the generator loading to prevent the shedding of high priority loads before all the potential power paths are evaluated. Since paralleling of generators not initially paralleled is not allowed, the only way to alleviate the situation is to shed load. This should be done on a priority basis by queuing up loads by priority for shedding. General sorting and queuing subroutines are available in the original GATOR code. Loads are shed one at a time with the usual calls to CSTAT, SYS_STATS, and CHKLOAD to redetermine the state of the system after each load shed. This is continues until all loads which can be powered are powered within generator capacities.

```
DO  30 I = 1, NUMBER OF SOURCES

/*  THE SOURCE IS OK IF OVRLD(I) .EQ. 0  */

      IF <OVRLD(I)  .EQ. 0> GOTO 30
      ELSE
            DO 40 J = 1, NUMBER OF LOADS
                   IF
                     <SRCPATH CHECK FROM SOURCE(OVRLD(I)) TO
                     ANY LOAD(J) INDICATES LOAD(J) IS UP> .AND.
                     <ALL OTHER COMPONENTS IN THE SRCPATH ARE UP>
                   THEN
                     <ADD THE LOAD(J) TO A PRIORITIZING QUEUE
                     BY COMPARING THE SHED PRIORITY IN
                     EQDATA(EQDATA(LOAD(J), 8), 4) TO THE
                     SAME PRIORITY FIELD FOR ANY OTHER BREAKERS
                     FOUND IN THIS LOOP>
      40    CONTINUE
      /*  SHED THE LOAD AT THE TOP OF THE QUEUE SINCE IT HAS
          THE LOWEST PRIORITY.  COULD CREATE A SECOND LEVEL
          CHECK TO FIND ANY LOADS WHICH HAVE EQUAL PRIORITY
```

```
AND SHED ONLY THE ONE WHICH WILL SOLVE THE
OVERLOAD CONDITION.  ANY LOAD JUST SHED WILL NOT
BE FOUND ON THE NEXT PASS BACK DOWN SINCE IT WILL
HAVE BEEN SHUT OFF BY THE LOAD SHED.  */

  <CALL CSTAT( LOAD(POINTED TO BY QUEUE), 1 )

   CALL SYS_STATS

/*  GO BACK TO CHKLOAD TO RE EVALUATE THE SYSTEM LOADING.
THIS SHOULD KEEP ON LOOPING UNTIL ALL OVERLOADS ON
THE GENERATORS HAVE BEEN SOLVED AND ALL LOADS WHICH
CAN BE POWERED ARE NOW POWERED.  */

   GOTO 10>
30  CONTINUE
```

Chapter 5. Analysis Method

The details of how a system survivability analysis is performed using BEAVER is presented with the use of an example. This chapter is intended to provide a detailed example to illustrate the formulation of an analysis. The system layout must be converted to a suitable form for input before then the program can be run to determine system survivability characteristics. For the example, a distribution system arrangement is converted from a one line diagram to a logic diagram format for analysis. The actual characteristics of the system are summarized in the **Survivability Concepts** chapter of this thesis, with figure 3 depicting the one line diagram.

Input File Formation

The formation of the input file consists of gathering parameter data, organizing the logical subgroups, and building the fault tree. A commented input file example is included in Appendix A. Most of the data fields are self explanatory with the exception of the specification of equipment parameters and the forr .ation of the fault tree structure. The first step is to convert the system arrangement into a form which can be input.

Figure 9 shows a simplified partial version of the conceptual naval shipboard electrical power distribution system architecture shown in figure 3.

Simplified Two Zone Diagram

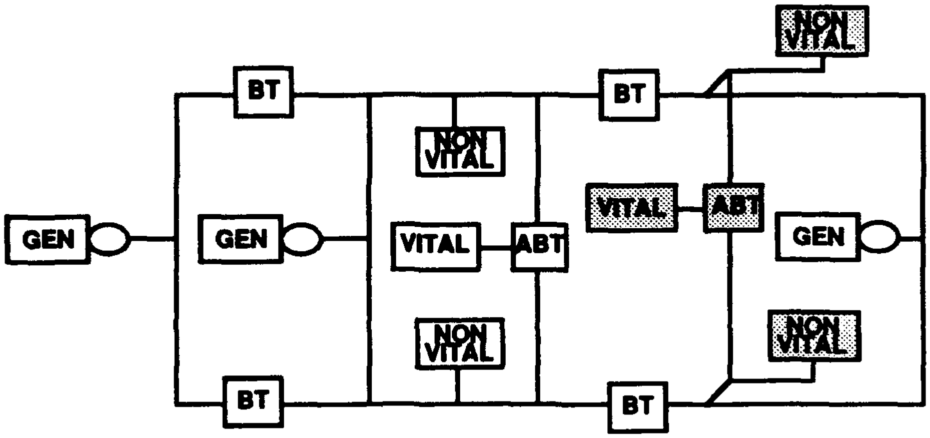

Figure 9. Simplified Conceptual Naval Shipboard Electrical Distribution System.

A System Implementation Diagram (SID) is formed from the one line diagram. It consists of identifying all individual pieces of equipment with a unique number label. This includes all power sources, loads, cables, and connection components such as switchboards, circuit breakers, and load centers. The SID for figure 9 is shown in detail in figures 10 through 13. The locations of all the components should be noted on the diagram for ease of translation into data input file form. The location data can be found by matching the equipment position relative to the watertight bulkheads from figure 3. The number labels on the figures indicate equipment numbers by type for reference on these drawings only. These numbers are not input to the input file.

The labels are abbreviations for the type of component. The shading indicates the level above baseline (ABL) for the components. The heavy dashed lines indicate watertight bulkheads, which coincide with zone boundaries in this layout.

ABT	Automatic Bus Transfer switch
B	Breaker
BT	Bus Tie breaker
C	Cable
G	Generator
L	Load
LC	Load Center cabinet

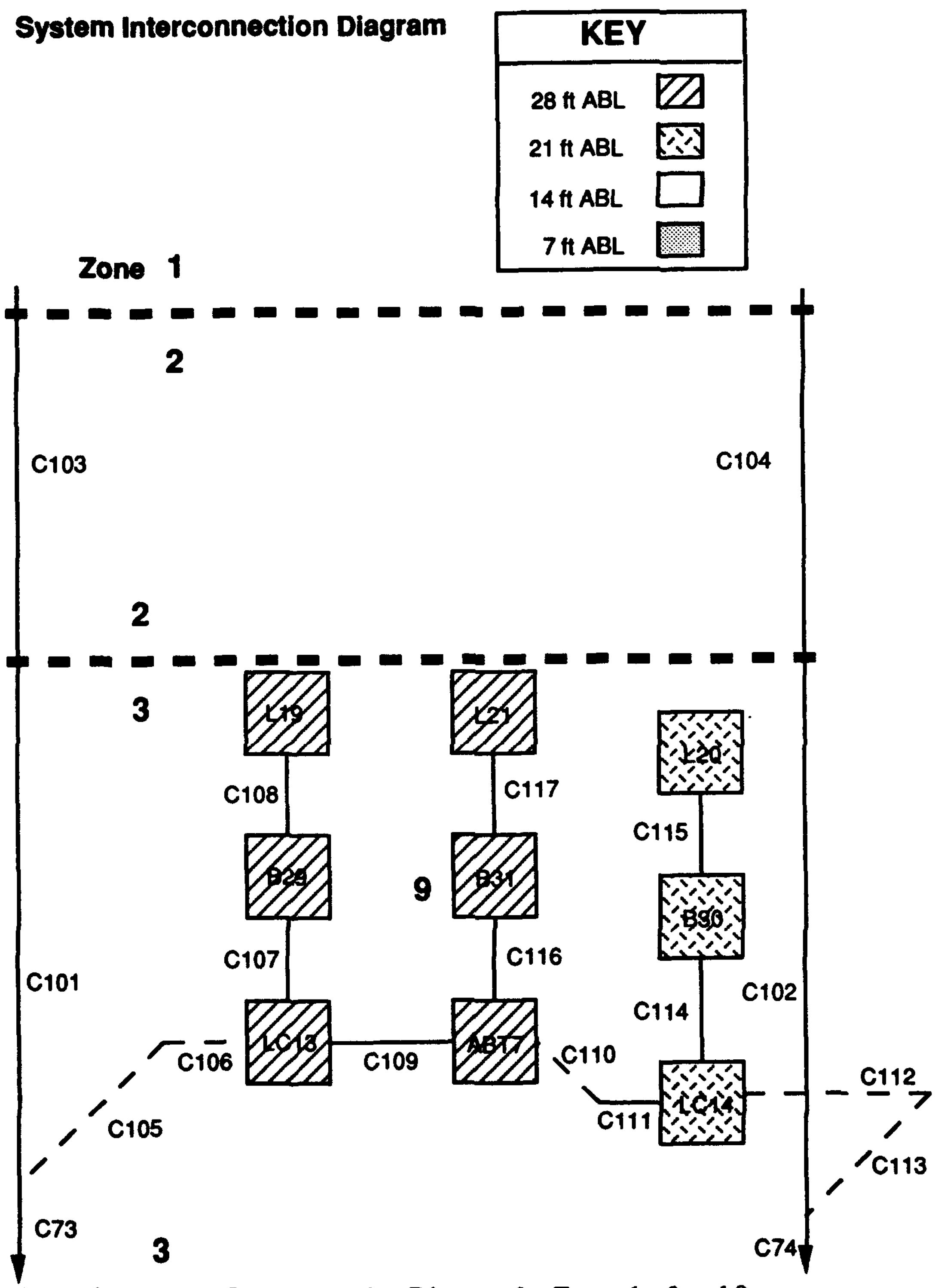

Figure 10. System Interconnection Diagram for Zones 1 - 3 and 9.

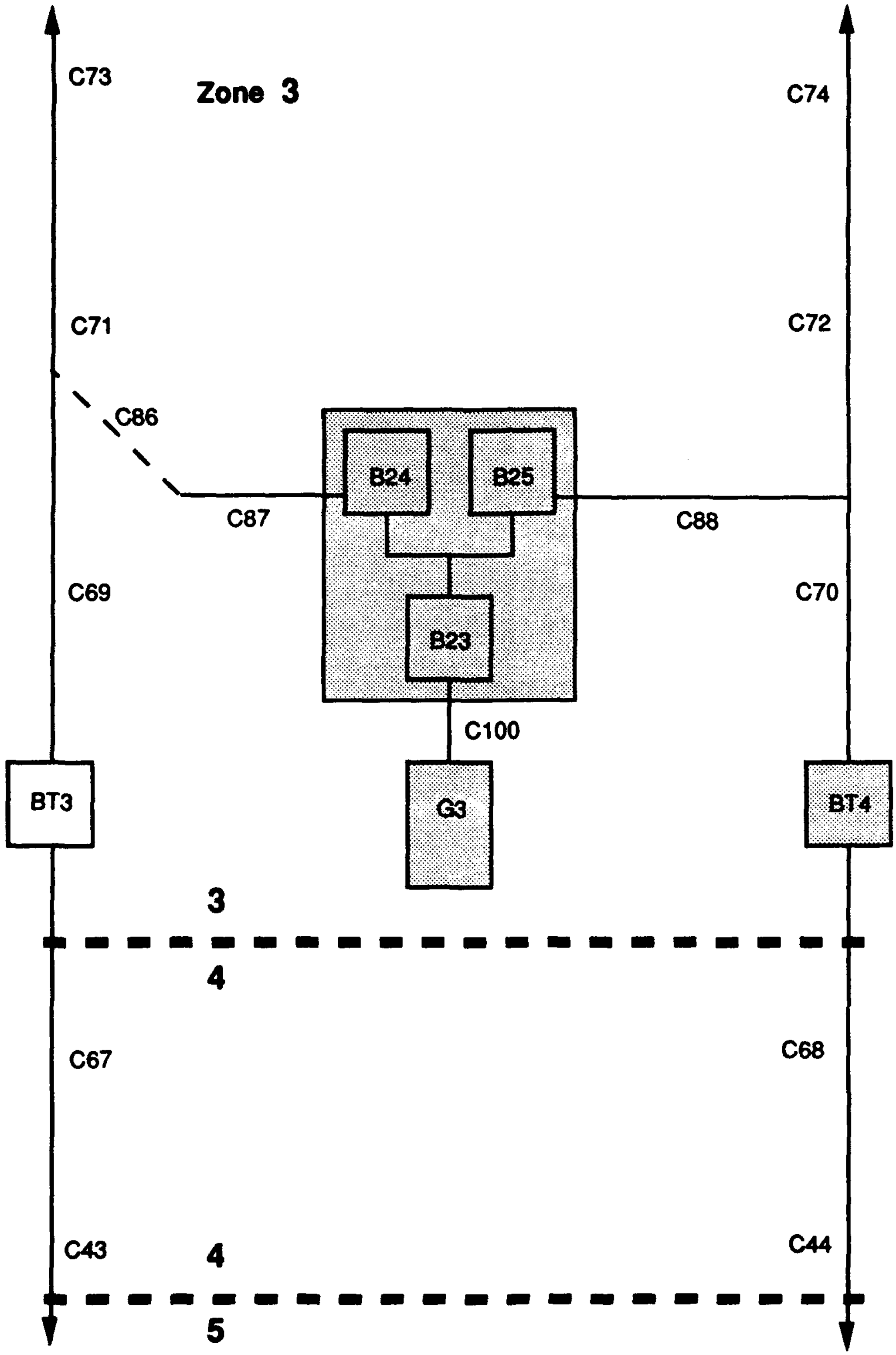

Figure 11. System Interconnection Diagram for Zones 3 - 5

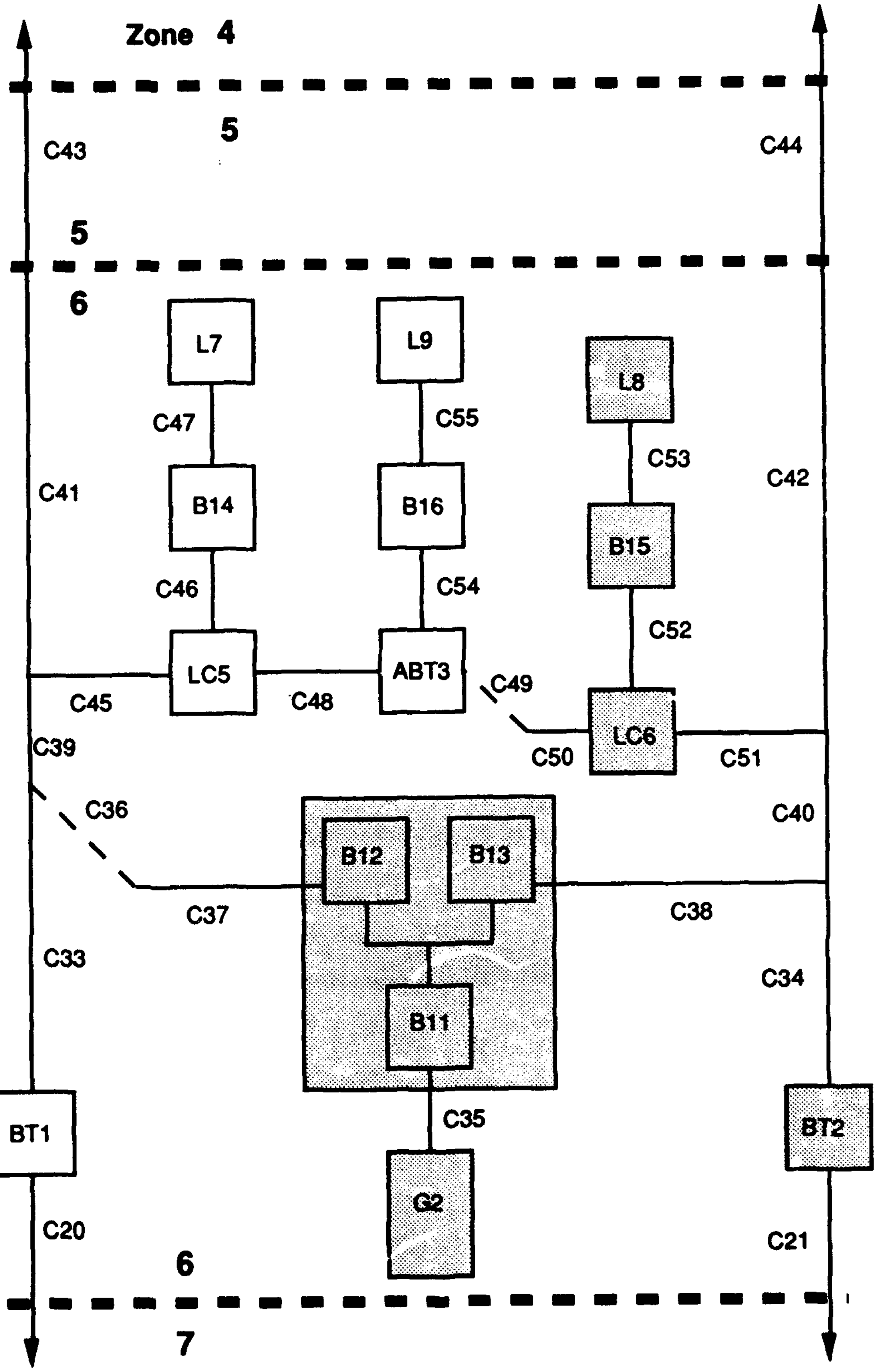

Figure 12. System Interconnection Diagram for Zones 5 - 7.

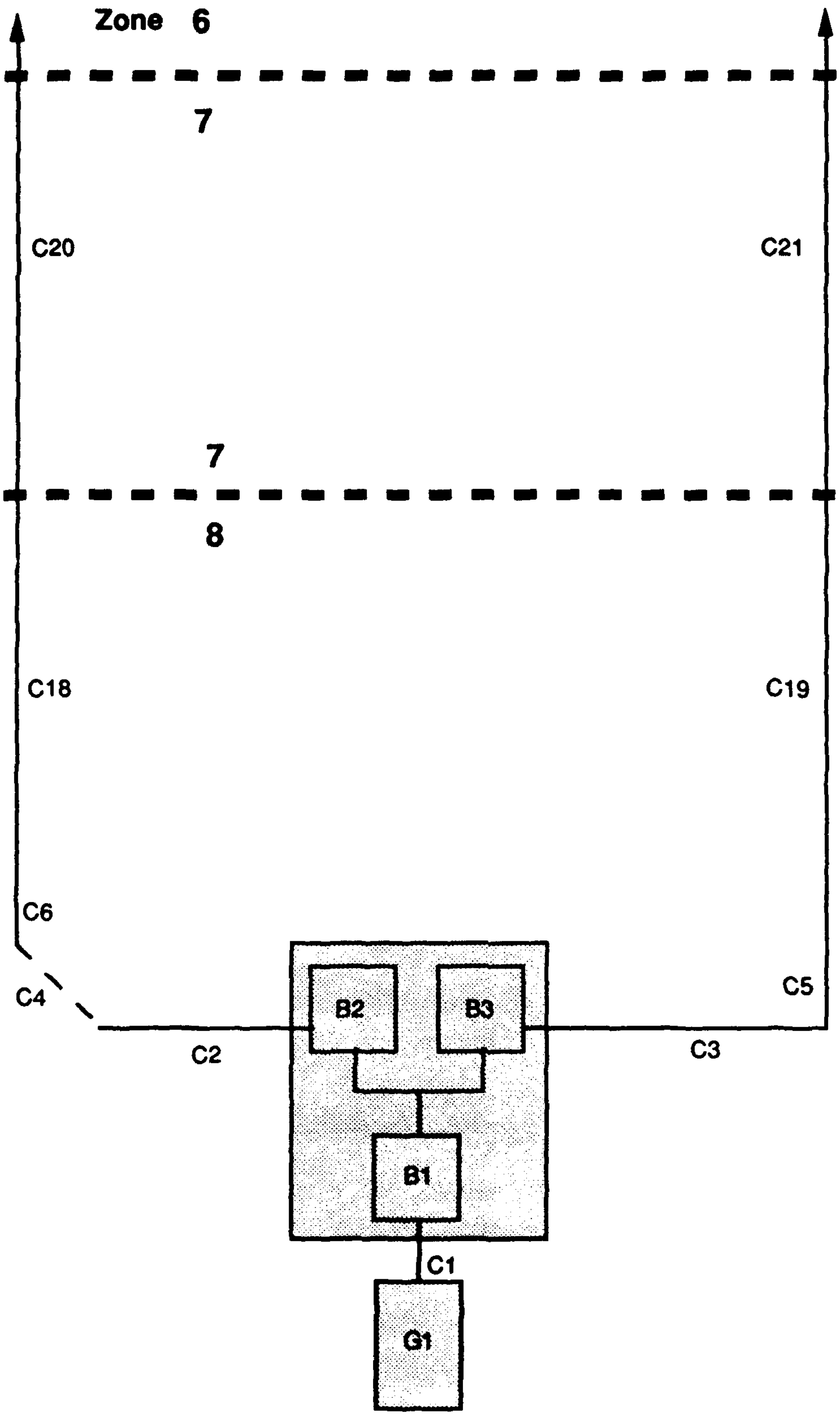

Figure 13. System Interconnection Diagram for Zones 7 & 8.

Equipment data specification

Equipment specification consists of identifying the characteristics of equipment types and assigning an equipment to a particular type. Data specification is entered in two places, the RMA data section at the top of the input file, and the survivability data section at the end of the input file. The division is necessary to allow commonality between GATOR and BEAVER program use. The characteristics of GATOR equipment types are input in the data section near the top of the input file. The data must be specified for a survivability analysis even if the RMA analysis is not performed. Here, the equipment type consists of assigning a number to each unique type along with a list of specific data including the type name, MTBF, MTTR, and repair priority.

For purposes of the example, the SID shown in figures 10 to 13 is used. The type 1 equipment named *power source* lists the characteristics of the generators. The individual generators identified as equipment numbers 1, 2, and 3 on the SID have their GATOR specification data listed in the type 1 assignment row of the input file. Similarly, equipment types 2 through 8 list characteristics for the other general types of equipment. Figure 14 illustrates that portion of the input file for a system with 8 different types of equipment.

Eq Type #	Eq Type Name	MTBF	MTTR
1	POWER SOURCE	025000.	6.
2	BREAKER	005000.	1.
3	LOAD CENTER	010000.	1.
4	BUS TIE	005000.	2.
5	FCS LOAD	003000.	2.
6	AUX LOAD	001000.	4.
7	ABT	003000.	3.
8	CABLE	008000.	2.

Figure 14. Partial Input File for Equipment Type Assignment.

An assignment section then maps all the individual equipment numbers to a specific type. Individual pieces of equipment are mapped to the type by specifying unique component numbers for each type of component. The list of numbers following the equipment type column indicates the equipment which have been assigned to that general type. The component numbers are the ones designated on the SID. Figure 15 shows the mapping for the 96 total pieces of equipment.

Eq Type #	Equipment Identification Number									
1	1	2	3							
2	4	5	6	7	8	9	10	11	12	13
2	14	15	16	17	18	93	94	95	96	
3	19	20	21	22						
4	23	24	25	26						
5	27	28	29							
6	30	31	32							
7	33	34								
8	35	36	37	38	39	40	41	42	43	44
8	45	46	47	48	49	50	51	52	53	54
8	55	56	57	58	59	60	61	62	63	64
8	65	66	67	68	69	70	71	72	73	74
8	75	76	77	78	79	80	81	82	83	84
8	85	86	87	88	89	90	91	92		

Figure 15. Partial Input File for Equipment-Type to Equipment-Number Mapping.

The BEAVER equipment specification data for the survivability analysis is input in the matrix at the bottom of the file. This matrix consists of 12 columns by as many rows as there are individual pieces of equipment, plus one row for the specification of the data input termination flag: "9999". The first column indicates the BEAVER equipment specification type, summarized in table 1. The BEAVER equipment specification types are different from the GATOR types listed above for the RMA data section input. The BEAVER types are included to allow the program to know what components are sources, loads, breakers, and cables since each must be processed differently.

Table 1. Equipment Type List	
Type	**Number**
Emergency power source	1
Power source	2
Equipment load	3
Breaker	4
Cable	5
Component cabinet	6
Bus tie	7
Automatic Bus Tie (ABT)	8

The second column is the individual equipment number as assigned previously in the RMA data section so that the program can refer back to the data listed there. The use of the next ten columns depends on the type of equipment specified. Table 2 summarizes data fields 3 - 8 for each type of equipment. Column 9 is used to indicate the level of blast protection, relative to the hit strength, to be placed on the component. It is a relative number based on the hit description in the format described in chapters 3 and 4. Column 12 is used to indicate the acquisition cost of the component. The costs are summed up at run time. The cable cost field is listed as a per length unit so that a total cost of the cable length is determined.

Columns 10 and 11 are used by equipment to indicate either a control breaker in the component path or a cascade for breaker control. For loads, the breaker in column 10 would be the breaker which controls power to the load. For sources, the column 10 breaker indicates the breaker which applies the power to the bus.

Table 2. EQDATA Array Equipment Type Data Field Summary				
Type Number	**Data Field Column Number**			
	3 - 5	**6**	**7**	**8**
1	x, y, z coordinates of equipment location	not used	not used	not used
2		max power	initial status	not used
3		load power	initial status	not used
4		priority	initial status	not used
5		x, y, z coordinates of other cable end point		
6		not used	not used	not used
7		not used	initial status	not used
8		not used	not used	not used

The ABT uses these fields to indicate which two breakers control the power from each bus. Note that the ABT breakers should be operated as an exclusive-or type of construction since there should never be power coming in simultaneously from two buses. This should be reflected in the initial status field of the EQDATA input for the breakers noted in the ABT fields. A user could program a control algorithm to enforce this during run time to ensure proper control rules for the ABT's are followed.

Breakers use these fields to indicate when a cascade effect is required for the control of the system. For example, if the generator goes down, then breaker indicated in column 10 should open to simulate the trip which should occur in a real system. When this breaker opens, the two port and starboard breakers pointed to by the main generator bus breaker should also be opened from a cascade effect.

Equipment Spares Policy

The input file key phrase "UNLIMITED SPARES" is necessary for the GATOR RMA analysis and indicates that each piece of equipment has unlimited spares on board for replacement. In general, the repair of defective parts is not performed unless the repair server is enabled. Spares policy is not important for survivability analysis unless the analysis is to move, via a phase shift, to a mission phase where RMA is performed and repair is allowed. This type of analysis might be desired if some estimated casualty repair time can be input as the MTTR for each piece of equipment, and the recovery of the system were to be simulated after a damage event as a time dependent analysis. Spare replacement is not performed in this analysis, nor was it tested in this version of BEAVER for use in phased trials.

Logical Interconnection Diagram

Once the system equipment has been identified and labelled, it must be arranged logically to show system interrelationships. Figure 16 depicts a simplified Logical Interconnection Diagram (LID). This type of drawing helps the user translate the information into the form necessary for reading into and evaluation by the program's fault tree structure.

Illustrative Example

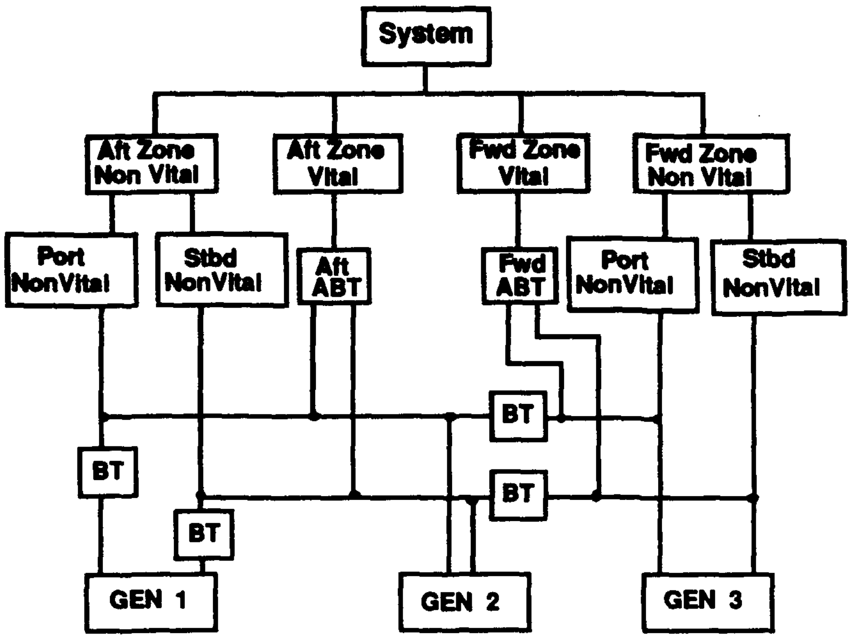

Figure 16. Logical Interconnection Diagram of Distribution System.

The system node is built by combining equipment into subgroups. The subgroups are then organized into larger subgroups until the top of the system is reached. The complete version of the LID created for input file conversion is shown in figures 17 to 19. Notice that in Figure 17 the system has been configured as 'survivable' only if all four subsystems--Aft Zone Non Vital, Aft Zone Vital, Fwd Zone Vital, and Fwd Zone Non Vital--have survived. Continuing downward, each subsystem has its own requirements in terms of smaller subgroups and equipment.

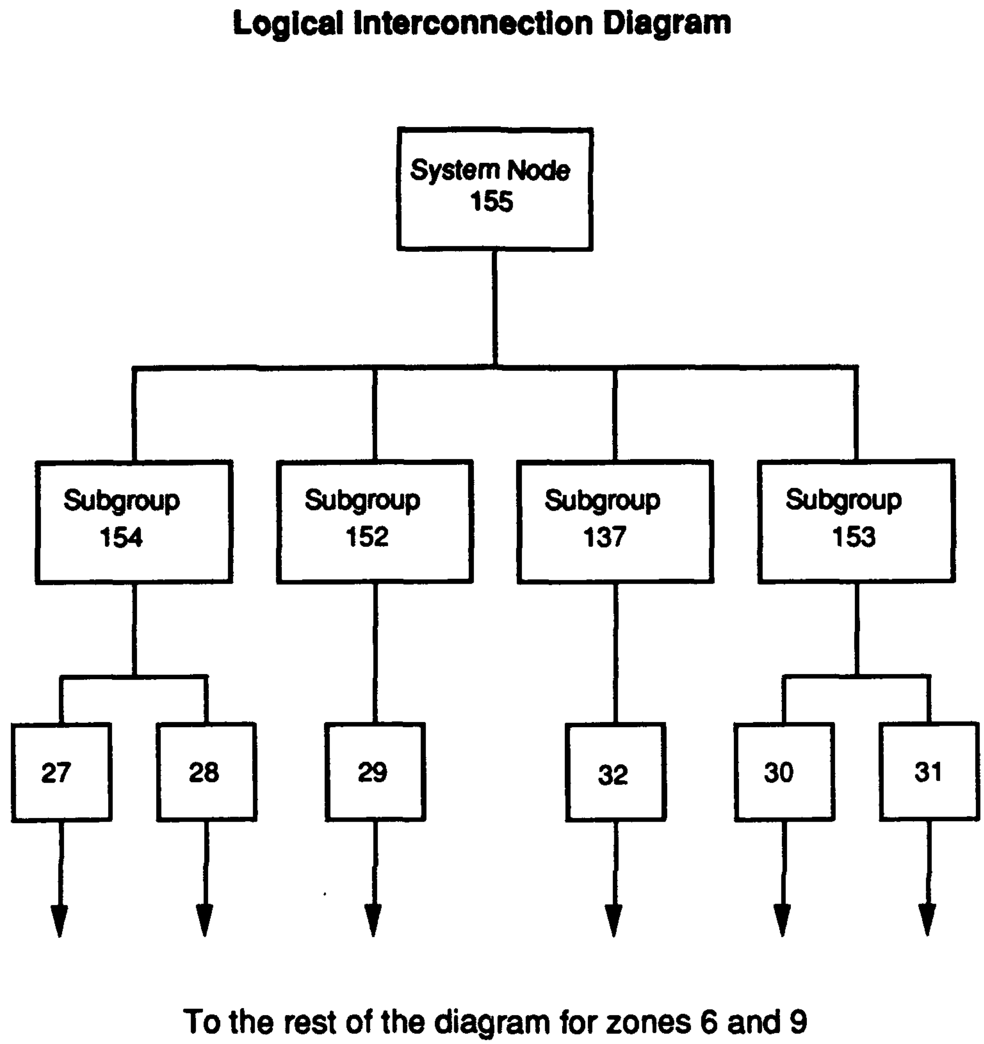

Figure 17. Top Level of the Logical Interconnection Diagram.

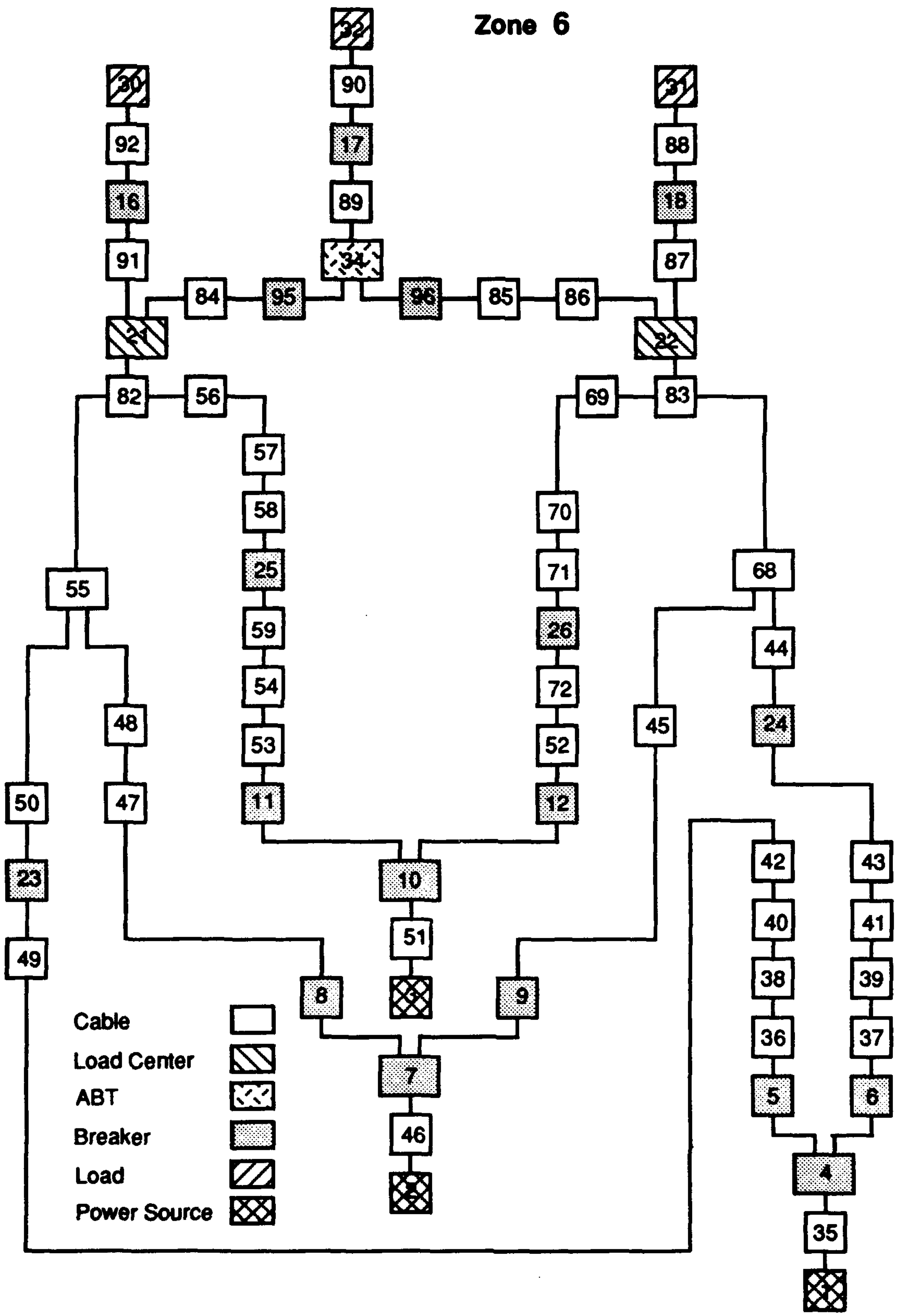

Figure 18. Logical Interconnection Diagram for Zone 6.

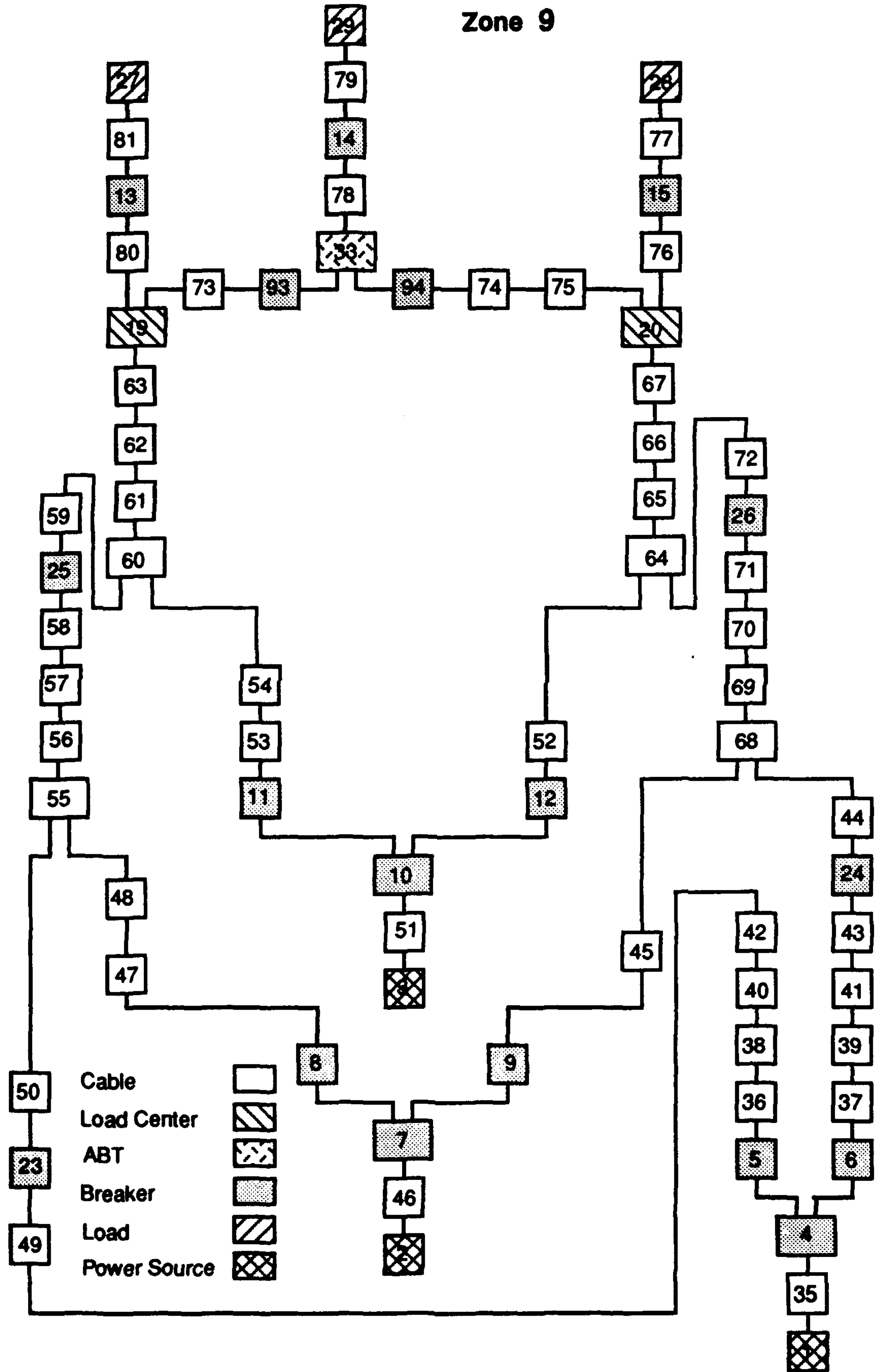

Figure 19. Logical Interconnection Diagram for Zone 9.

Fault Tree Formation

Once each piece of equipment is identified with a unique number and mapped to an equipment type, the system is formed by organizing the components into subgroups. The subgroup specification section begins in the section following the equipment-to-type map, immediately following the key phrase "SYST". The number following the SYST key word is the system number. This number is significant in that it tells the program the number of the final system block at the top of the tree. This number is determined after the tree is formed.

The subgroup specification matrix consists of as many rows as there are subgroups. The number of columns in the matrix depends on the number of components in each subgroup. Subgroups should be organized either as logical groups of individual components which are to be tracked together for data information or as logical groups for convenience of building up the tree. The first column indicates how many of the items listed in the right hand columns 3 - 13 of the same row, must be active for the survival of the subgroup. The second column is the subgroup number. The right hand columns 3 - 13 are the individual component numbers or subgroup numbers which are grouped into the subgroup number indicated in the second column.

The tree is formed in the input file from the user's translation of the LID. The system is built by combining equipment into subgroups. The subgroups are then organized into larger subgroups until the top of the system is reached. An example input list is reproduced in figure 20 to show how these requirements are listed in the input file. The full list is in the input file in Appendix A.

```
SYST      155
   3 100    4  35   1
   3 101    7  46   2
   3 102   10  51   3
   3 103   39  37   6
   4 104   40  38  36   5
   2 105   45   9
   3 106   48  47   8
   2 107   52  12
   3 108   54  53  11
   2 109  103 100
   2 110  104 100
   2 111  105 101
   2 112  106 101

   5 145   27  81  13  80 144
   5 146   20  67  66  65 143
   5 147   28  77  15  76 146
   3 148  144  73  93
   4 149  146  75  74  94
   1 150  148 149
   2 151   33 150
   5 152  151  78  14  79  29
   1 153  131 132
   1 154  145 147
   4 155  153 154 137 152
```

Figure 20. Partial Logical System Structure as Listed in the Input File.

In this figure, the system is #155, and every other intermediate subgroup is some number between 100 and 154. The leftmost integer in each line is the subgroup requirement ; e.g., Subgroup #100 requires three equipments numbered as 1, 35, and 4, whereas Subgroup #150 requires only 1 of two smaller subgroups 148 or 149. The 3 of 3 indicated for subgroup # 100 indicates a series connection of equipments numbered 1, 35, and 4. The 1 of 2 requirement for subgroup 150 for subgroups numbered 148 and 149 implies a parallel relationship between the parent, subgroup 150, and the children, subgroups 148 and 149.

The ordering of items, either components or subgroups, in columns 3 - 13 is important when sequentiality of the components is to be formed in the POWERPATH subroutine. The tree components and subgroups are read in and investigated from the leftmost column to the rightmost column in the subgroup row. When the LID structure is input to this data section, the rightmost columns should contain the items at the bottom of the subgroup structure. For example, the ordering of subgroup #100 is such that the power source is at the very bottom of the tree branch. The sequential ordering of the component listing should be the power source #1 to the right, the cable # 34 to its left, and the breaker #4 in the leftmost column, column number 3 of the list. If a subgroup is to be included in with a string of components, the subgroup should go in the rightmost column. The ordering of parallel subgroups does not matter.

The entire system is made up of these series and parallel connections all the way from the bottom, where individual pieces of equipment make up the leaves, to the very top, where the system node is made up of connections of the lower subgroups, as branches, into one final goal.

Control Implementation

In order to implement control algorithms, the user must encode the strategy into the BEAVER FORTRAN source code. The variability and complexity of control schemes is such that a generic control programming capability is neither easily implemented nor particularly desirable. The POWERCHK pseudo coded example in Chapter 4 can be used as a guide for the use of some of the data structures provided in the basic BEAVER code. The development of a "library" of basic potential control schemes may eventually be useful, but most implementations are rather specific and require unique coding to simulate the control.

Program Operation

For deterministic survivability analysis, the program runs through once and outputs the system status down to the equipment level. For the Monte Carlo type analysis, the program loops through a specified number of times and outputs a percentage survivability result down to the equipment level. For coverage of the entire ship, either of the two analysis methods require moving the hit location around manually over multiple analyses.

Figure 21 gives a sample of some of the survivability output information provided by BEAVER. The full output file is located in Appendix B. In this case, Monte Carlo sampling was invoked to provide the damage profile, and five trials were run.

Partial Equipment Survivability Output for Single Trial

```
LOAD     5, EQUIPMENT #    31 DAMAGED.
LOAD     6, EQUIPMENT #    32 NOT POWERED.
LOAD    32 NOT POWERED, BUT IT SHOULD BE.
SOURCE(    3) IS OVERLOADED BY    600.00 KW.
```

Partial Survivability Output File for Multiple Trials

```
COMPONENT/BLOCK SURVIVABILITY (IN PERCENT)
FOR     5 SURVIVABILITY TRIALS.

BLOCK #     1:                                   100.00 PERCENT.
BLOCK #     2:                                   100.00 PERCENT.
BLOCK #     3:                                   100.00 PERCENT.
BLOCK #    27:                                   100.00 PERCENT.
BLOCK #    29:                                   100.00 PERCENT.
BLOCK #    30:                                   100.00 PERCENT.
BLOCK #    31:                                      .00 PERCENT.
BLOCK #    32:                                   100.00 PERCENT.
BLOCK #   101:                                    40.00 PERCENT.
BLOCK #   102:                                   100.00 PERCENT.
BLOCK #   103:                                   100.00 PERCENT.
BLOCK #   137:                                      .00 PERCENT.
BLOCK #   153:                                    20.00 PERCENT.
BLOCK #   154:                                   100.00 PERCENT.
BLOCK #   155:                                      .00 PERCENT.
```

Figure 21. Sample Partial BEAVER Output Files.

The first part of the output shows the results of the system checks from POWERCHK. Had the reconfiguration pseudo code from Chapter 4 been implemented, the load to source power path continuity would have been checked first to determine if any ABT switching would have reconnected load 32. If not, then the load would remain unpowered. The code could also have output where the faults occurred to assist in damage isolation. The algorithm would then proceed to check the overload of generator number 3 to find the loads to shed to correct the condition. The load priority would have found the loads to shed one at a time until the overload cleared. System survivability would have been improved if some of the loads could have been powered from alternate paths without overloading the generators. The load survivability in this case would be improved since load shed is now controlled by a connection priority determined by the system rather than by waiting for circuit overloads to trip breakers on protective action.

The second part of the output gives a survivability index for each subgroup of the system as both a percent survivability and a percent survivability divided by the total system cost. Since the system node #155 requires 4 of 4 subsystems 137, 152, 153, and 154 to survive, it has 0.00 % survivability due to subgroup 137 being down for all trials. Investigation of the output file along with a tree search would show that equipment which caused subgroup 137 to be down. Multiple runs could be made at different shielding levels to determine if this would help to mitigate the problem. Relocation of critical equipment could also help. These numbers would have to be summed over many hit trials around the ship to determine an overall survivability index.

The cost indices are shown in figure 22. These were obtained by dividing the survivability percentage by the total cost of the system, then multiplying by 10,000 to bring the numbers up to a better range to work with. Since the highest survivability is 100%, the 6.8 index is the best a subsystem or component can do. Any number less than this is relatively less effective in providing survivability with zero being the worst case. As in the percentage indices, the actual cost indices should be summed over a distribution of hits around the ship to determine system survivability. The numbers obtained here for cost indices would have to be compared to other systems since they provide only a relative index.

```
EQUIPMENT #      1     SURV COST INDEX:       6.8
EQUIPMENT #      2     SURV COST INDEX:       6.8
EQUIPMENT #      3     SURV COST INDEX:       6.8
EQUIPMENT #      8     SURV COST INDEX:       2.7
EQUIPMENT #      9     SURV COST INDEX:       2.7
EQUIPMENT #     10     SURV COST INDEX:       6.8
EQUIPMENT #     31     SURV COST INDEX:        .0
EQUIPMENT #     32     SURV COST INDEX:       6.8
GROUP #        101     SURV COST INDEX:       2.7
GROUP #        102     SURV COST INDEX:       6.8
GROUP #        103     SURV COST INDEX:       6.8
GROUP #        137     SURV COST INDEX:        .0
GROUP #        152     SURV COST INDEX:       6.8
GROUP #        153     SURV COST INDEX:       1.4
GROUP #        154     SURV COST INDEX:       6.8
GROUP #        155     SURV COST INDEX:        .0
```

Figure 22. Survivability Cost Index Output.

Combat effectiveness indices could be interpreted from the subsystem survivability indices. For example the indices for the four subgroups making up the system node 155 could be defined as indicating propulsion and combat systems powering.

Load	Zone	Subgroup
Propulsion Vital	6	137
Propulsion Non Vital	6	153
Combat Systems Vital	9	152
Combat Systems Non Vital	9	154

The loss of these systems, as indicated by the survivability indices, could be given a weighting factor in a figure of merit calculation to determine a separate index for comparison to other systems. The weighting would account for the importance of the subsystem power loss to various aspects of combat capability such as mobility or offensive and and defensive capabilities. The weighting assigned and the effect on combat capability would have to be determined by the warfare specialists interpreting the survivability results.

Chapter 6. Outstanding Issues and Conclusions

The benefits of distributed systems will only become obvious when the entire system and its control are considered as an entire working unit. The static survivability analyses performed on many current systems will not be able to effectively evaluate the merits of advanced controlled systems, especially those controlled by expert systems and automated controllers. Development of tools such as BEAVER will become more necessary as integrated control architecture distributed systems of all types from electrical distribution to normal ship operation and damage control are incorporated into new designs.

Another benefit from this type of combined system and controls evaluation capability is the ability to analyze the entire system as it will be implemented. Approaching optimal system design will become more efficient when the operational capabilities can be simulated rather than depending on worst case scenarios and high end estimates of performance. This capability would also allow the coordinated design and evaluation of systems and their controls from the outset rather than designing the system and placing the controls around to connect the parts together.

Outstanding Issues

The aspects of reconfiguration and control simulation for the BEAVER program have only just begun and are still under development. In general, any specific control rules and algorithms desired would have to be encoded in FORTRAN and added to the existing source code for a use in a particular system since broad generalities in control system implementation are not common. The BEAVER program does not need the reconfiguration capability to perform either the survivability or reliability analyses, but the overall program structure is present to allow incorporation of different control algorithms.

The RMA analysis capabilities of the GATOR program are enhanced with the addition of the ability to turn off equipment before analysis begins for two reasons. First, the system evaluator can now input the system once as a logical construction and simply line up the equipment for the given analysis. The current GATOR implementation requires reconfiguring the entire input file to accomplish analysis of different configurations of the same basic system. The second capability is the ability to perform conditional reliability analysis with given equipment in a failed state. This type of analysis is useful for "what if" analysis of systems in a degraded state to begin with.

The incorporation of the reconfiguration capability to the RMA analysis would provide useful results in analyzing complete systems including the controls. The issues which differ from the survivability assessment are the repair and reconfiguration of equipment depending on longer term rules and operational phases. Not only would control systems have to be modeled, but perhaps operational rules as well.

Conclusions

The BEAVER program provides a convenient and efficient analysis environment for evaluation of distributed system reliability and survivability at the conceptual design level, including the ability to incorporate the evaluation of the control system. The inclusion of an affordability analysis capability provides data necessary to perform cost indexing of alternative system arrangements. Benefits of this methodology and approach *include:*

- Allows evaluation of systems complete with control as implemented in the actual system configuration.
- Conceptual level design tool which allows early design level trade offs.
- Ship geometry independent to reduce analysis overhead.
- Applicable to any distributed system for ship design process flexibility and commonality.
- PC based operation for user convenience.

References

[1] IEEE Tutorial Course, Reliability Assessment of Composite Generation and Transmission Systems, Course Text 90EH0311-1-PWR, New York, 1989.

[2] IEEE Tutorial Course, Distribution Automation, Course Text 88EH0280-8-PWR, New York, 1988.

[3] Laverki, E. and E. J. Holmes, Electricity Network Distribution Design, IEE Power Engineering Series 9, Peter Peregrinus, Ltd., London, UK, 1989.

[4] Gonen, T., Electric Power Distribution System Engineering, McGraw-Hill, New York, NY, 1986.

[5] McCormick, N., Reliability and Risk Analysis, Academic Press, New York, NY, 1981.

[6] 10th International Conference on Electricity Distribution, Subject Area: 4. Operation and Control in Public Supply Systems, IEE Conference Publication No 305, 1989.

[7] 10th International Conference on Electricity Distribution, Subject Area: 6. Design and Planning of Public Supply Systems, IEE Conference Publication No 305, 1989.

[8] Tinney, W. and C. E. Hart, *Power Flow Solution by Newton's Method*, IEEE Transactions on Power Apparatus and Systems, Volume PAS-86, No. 11, November 1967.

[9] Luenberger, D., Introduction to Dynamic Systems, Wiley and Sons, New York, NY, 1979.

[10] Larsen, R. and M. L. Marx, An Introduction to Mathematical Statistics and its Applications, Prentice-Hall, Inc., Englewood Cliffs, NJ, 1981.

[11] Wertz, J. and W. J. Larson, Space Mission Analysis and Design, Kluwer Academic Publishers. Norwell, MA, 1991.

[12] Stagg and El-Abiad, Computer Methods in Power System Analysis

[13] Stevenson, W. D., Elements of Power System Analysis, McGraw-Hill, 1982.

[14] Bergen, A. R., Power Systems Analysis, Prentice-Hall, Inc., 1986.

[15] Debs, A., Power System Operation and Control, Kluwer Academic Publishers, 1987.

[16] Windsor, G. B., The Fundamentals of Ship Survivability Analysis and Design, Naval Postgraduate School Thesis, March, 1989.

[17] David Taylor Research Center, SVM User's Manual (Confidential), SVM Version 10, 1984.

[18] Fulton, J. P., Introduction of the Fast SVM (Confidential), David Taylor Research Center, 1988.

[19] Fairhead, D. L. and S. L. Cohen, *A Computer Model for Damage Tolerance Analysis*, Naval Engineers Journal, September, 1988.

[20] Allinger, D. F., and A. L. Schor, Enhancements to the GATOR Large System Reliability Estimator Version 2.0 (Not for public release), CSDL-R-2106, Charles Stark Draper Laboratory, December 1988.

Appendix A. Input Files

COMMENTED BEAVER INPUT FILE: SOME DATA REMOVED TO SHORTEN THE FILE.

NOTE: THESE COMMENTS ARE FOR EDIFICATION ONLY AND CAN NOT APPEAR IN ANY ACTUAL FILE.

THE FIRST ENTRY IS THE SURVIVABILITY ANALYSIS SWITCH "SURVIVE".
SURVIVE = 1 SURVIVABILITY ANALYSIS IS PERFORMED. SURVIVE = 0 RMA ANALYSIS IS PERFORMED.

```
 1
```

THE NEXT LINE IS THE ANALYSIS RUN IDENTIFIER (NAME) AS READ BY THE PROGRAM.

```
     ELECTRICAL DISTRIBUTION SYSTEM SURVIVABILITY 5/7/92
```

INPUT PARAMETERS FOR:

1. NUMBER OF MISSIONS
2. NUMBER OF MISSIONS BETWEEN PRINTOUT OF DATA TO OUTPUT FILE (FOR RMA ONLY)
3. LOWEST GROUP NUMBER

NOTE: THE LOWEST GROUP NUMBER DOES NOT NEED TO BE THE NEXT NUMBER AFTER THE HIGHEST EQUIPMENT NUMBER, BUT THE EQUIPMENT NUMBERS USED TO IDENTIFY INDIVIDUAL EQUIPMENT CANNOT EXCEED THE LOWEST GROUP NUMBER.

4. NUMBER OF STANDARD DEVIATIONS (RMA ONLY)
5. NUMBER OF PHASE TYPES (ALWAYS 1 FOR SURVIVABILITY ANALYSIS)
6, 7, 8, 9. FOUR SEEDS FOR THE RANDOM NUMBER GENERATOR

```
00005 0005  100 1.96   1   113524955  1332347639  1332347639   113524955
```

THE NEXT LINE IS FOR RMA PHASE IDENTIFICATION. THERE SHOULD ALWAYS BE AT LEAST A 1 IN THE FIRST COLUMN FOR SURVIVABILITY ANALYSIS SINCE THERE IS ONLY ONE PHASE. THE SECOND NUMBER IS THE DURATION OF THE PHASE. IT IS USED FOR RMA ONLY.

```
 1  00100.
```

THIS NEXT NUMBER IS THE PRINTOUT OPTION. IT IS ONLY FOR RMA ANALYSIS.

```
   4
```

THE NEXT LINE IS FOR THE REPAIR SERVER. A 1 TURNS THE SERVER OFF, A 0 TURNS THE SERVER ON. THE SERVER SHOULD BE LEFT OFF FOR SURVIVABILITY ANALYSIS.

```
   1
```

THIS NEXT LINE IS FOR THE NUMBER OF STAGES IN THE ERLANG SERVICE DISTRIBUTION (RMA ONLY).

```
                      1
```

THIS NEXT SET OF DATA IS DESCRIBED IN CHAPTER 5.

1. EQUIPMENT TYPE
2. EQUIPMENT NAME
3. MTBF
4. MTTR
5. DUTY CYCLE FACTOR (RMA ONLY)
6. REPAIR PRIORITY (RMA ONLY)

```
   1 POWER SOURCE       025000.  6.  1.  1
   2 BREAKER            005000.  1.  1.  1
   8 CABLE              008000.  2.  1.  1
```

EQUIPMENT TYPE TO EQUIPMENT NUMBER MAPPING AS DESCRIBED IN CHAPTER 5.

```
   1   1   2   3
   2   4   5   6   7   8   9  10  11  12  13
   2  14  15  16  17  18  93  94  95  96
   3  19  20  21  22
   4  23  24  25  26
   5  27  28  29
   6  30  31  32
   7  33  34
   8  35  36  37  38  39  40  41  42  43  44
   8  45  46  47  48  49  50  51  52  53  54
   8  55  56  57  58  59  60  61  62  63  64
   8  65  66  67  68  69  70  71  72  73  74
   8  75  76  77  78  79  80  81  82  83  84
   8  85  86  87  88  89  90  91  92
```

UNLIMITED SPARES KEY WORD FOR RMA ANALYSIS.

```
UNLIMITED SPARES
```

SYSTEM INTERCONNECTION DEFINITION LIST. DESCRIBED IN CHAPTER 5.

```
SYST      155
   3 100   4  35   1
   3 101   7  46   2
   3 102  10  51   3
   3 103  39  37   6
   4 104  40  38  36   5
   1 153 131 132
   1 154 145 147
   4 155 153 154 137 152
```

EQDATA DATA ARRAY AS DESCRIBED IN CHAPTER 5.

```
  2    1  450.00    0.00    10.00 1800.00    0.00    0.00   50.00    4.0     0.00 5550.00
  2    2  290.00    0.00    10.00 1800.00    0.00    0.00   50.00    7.0     0.00 5550.00
  3   28  100.00   10.00    30.00 1400.00    0.00    0.00   50.00   15.0     0.00 7900.00
  3   29  100.00    0.00    40.00  400.00    0.00    0.00   80.00   14.0     0.00 6700.00
  5   92  255.00  -10.00    20.00  260.00  -10.00   20.00   60.00   0.00     0.00   50.00
9999
```

HIT DATA FOR DAMAGE ROUTINE.

1. X LOCATION
2. Y LOCATION
3. Z LOCATION
4. MINIMUM RADIUS OF HIT DESIRED.
5. MAXIMUM RADIUS OF HIT DESIRED.
6. DIMENSIONLESS HIT BURST MAGNITUDE.
7. HIT DECAY FACTOR TO ADJUST HIT RANGE PROFILE.

```
  260.00   20.00   25.00   28.00   38.00  180.00    0.05
```

THESE NEXT FIELDS INDICATE THE SUBSYSTEMS DESIRED FOR ANALYSIS OUTPUT. A BLANK LINE INDICATES THAT ALL COMPONENT AND GROUP INFORMATION IS TO BE OUTPUT.

1 2 3 27 28 29 30 31 32 137 152 153 154 155

FOR FORMATTING INFORMATION FOR THE ACTUAL INPUT FILE, REFER TO THE EXAMPLE FILE PROCEEDING THIS PAGE, OR CHECK THE READIT SUBROUTINE CODE IN THE BEAVER SOURCE FILE.

```
1
    ELECTRICAL DISTRIBUTION SYSTEM SURVIVABILITY 5/7/92
00005 0005  100 1.96   1   113524955  1332347639  1332347639   113524955
 1  00100.

   4
   1
                          1
   1 POWER SOURCE         025000.  6.  1.  1
   2 BREAKER              005000.  1.  1.  1
   3 LOAD CENTER          010000.  1.  1.  1
   4 BUS TIE              005000.  2.  1,  1
   5 FCS LOAD             003000.  2.  1.  1
   6 AUX LOAD             001000.  4.  1.  1
   7 ABT                  003000.  3.  1.  1
   8 CABLE                008000.  2.  1.  1

   1   1   2   3
   2   4   5   6   7   8   9  10  11  12  13
   2  14  15  16  17  18  93  94  95  96
   3  19  20  21  22
   4  23  24  25  26
   5  27  28  29
   6  30  31  32
   7  33  34
   8  35  36  37  38  39  40  41  42  43  44
   8  45  46  47  48  49  50  51  52  53  54
   8  55  56  57  58  59  60  61  62  63  64
   8  65  66  67  68  69  70  71  72  73  74
   8  75  76  77  78  79  80  81  82  83  84
   8  85  86  87  88  89  90  91  92

UNLIMITED SPARES
SYST       155
   3 100   4  35   1
   3 101   7  46   2
   3 102  10  51   3
   3 103  39  37   6
   4 104  40  38  36   5
   2 105  45   9
   3 106  48  47   8
   2 107  52  12
   3 108  54  53  11
   2 109 103 100
   2 110 104 100
   2 111 105 101
   2 112 106 101
   2 113 107 102
   2 114 108 102
   4 115  44  24  43  41
   4 116  50  23  49  42
   2 117 115 109
   2 118 116 110
   1 119 111 117
   1 120 112 118
   2 121  68 119
   2 122  55 120
   5 123  56  57  58  25  59
   5 124  69  70  71  26  72
   2 125 124 113
   2 126 123 114
   1 127 122 126
   1 128 121 125
   2 129  82 127
```

2	130	83	128				
6	131	30	92	16	91	21	129
6	132	31	88	18	87	22	130
4	133	95	84	21	129		
5	134	96	85	86	22	130	
1	135	133	134				
2	136	34	135				
5	137	32	90	17	89	136	
2	138	123	122				
1	139	114	138				
2	140	60	139				
2	141	124	121				
1	142	113	141				
2	143	64	142				
5	144	19	63	62	61	140	
5	145	27	81	13	80	144	
5	146	20	67	66	65	143	
5	147	28	77	15	76	146	
3	148	93	73	144			
4	149	94	74	75	146		
1	150	148	149				
2	151	33	150				
5	152	29	79	14	78	151	
1	153	131	132				
1	154	145	147				
4	155	153	154	137	152		

2	1	450.00	0.00	10.00	1800.00	0.00	0.00	50.00	4.0	0.00	5550.00
2	2	290.00	0.00	10.00	1800.00	0.00	0.00	50.00	7.0	0.00	5550.00
2	3	165.00	0.00	10.00	1800.00	0.00	0.00	50.00	10.0	0.00	5550.00
3	27	100.00	-10.00	40.00	600.00	0.00	0.00	50.00	13.0	0.00	8900.00
3	28	100.00	10.00	30.00	1400.00	0.00	0.00	50.00	15.0	0.00	7900.00
3	29	100.00	0.00	40.00	400.00	0.00	0.00	80.00	14.0	0.00	6700.00
3	30	255.00	-10.00	20.00	650.00	0.00	0.00	60.00	16.0	0.00	5600.00
3	31	255.00	10.00	10.00	850.00	0.00	0.00	60.00	18.0	0.00	3900.00
3	32	255.00	0.00	20.00	400.00	0.00	0.00	80.00	17.0	0.00	9800.00
4	4	370.00	0.00	10.00	0.00	0.00	0.00	30.00	5.0	6.0	600.00
4	5	370.00	0.00	10.00	0.00	0.00	0.00	30.00	0.00	0.00	430.00
4	6	370.00	0.00	10.00	0.00	0.00	0.00	30.00	0.00	0.00	430.00
4	7	280.00	0.00	10.00	0.00	0.00	0.00	30.00	8.0	9.0	500.00
4	8	280.00	0.00	10.00	0.00	0.00	0.00	30.00	0.00	0.00	430.00
4	9	280.00	0.00	10.00	0.00	0.00	0.00	30.00	0.00	0.00	430.00
4	10	145.00	0.00	10.00	0.00	0.00	0.00	30.00	11.0	12.0	500.00
4	11	145.00	0.00	10.00	0.00	0.00	0.00	30.00	0.00	0.00	430.00
4	12	145.00	0.00	10.00	0.00	0.00	0.00	30.00	0.00	0.00	430.00
4	13	105.00	-10.00	40.00	0.00	0.00	0.00	30.00	0.00	0.00	630.00
4	14	105.00	10.00	40.00	0.00	0.00	0.00	30.00	0.00	0.00	630.00
4	15	105.00	0.00	30.00	0.00	0.00	0.00	30.00	0.00	0.00	630.00
4	16	260.00	-10.00	20.00	0.00	0.00	0.00	30.00	0.00	0.00	630.00
4	17	260.00	0.00	20.00	0.00	0.00	0.00	30.00	0.00	0.00	630.00
4	18	260.00	10.00	10.00	0.00	0.00	0.00	30.00	0.00	0.00	630.00
6	19	120.00	-10.00	40.00	0.00	0.00	0.00	90.00	0.00	0.00	1250.00
6	20	120.00	10.00	30.00	0.00	0.00	0.00	90.00	0.00	0.00	1250.00
6	21	265.00	-10.00	20.00	0.00	0.00	0.00	90.00	0.00	0.00	1250.00
6	22	265.00	10.00	10.00	0.00	0.00	0.00	90.00	0.00	0.00	1250.00
7	23	290.00	-14.00	20.00	0.00	0.00	0.00	50.00	0.00	0.00	3250.00
7	24	290.00	14.00	10.00	0.00	0.00	0.00	50.00	0.00	0.00	3250.00
7	25	165.00	-14.00	20.00	0.00	1.00	0.00	50.00	0.00	0.00	3250.00
7	26	165.00	14.00	10.00	0.00	1.00	0.00	50.00	0.00	0.00	3250.00
8	33	120.00	0.00	40.00	0.00	0.00	0.00	30.00	93.0	94.0	4000.00
8	34	265.00	0.00	20.00	0.00	0.00	0.00	30.00	95.0	96.0	4000.00
4	93	120.00	0.00	40.00	0.00	0.00	0.00	40.00	0.00	0.00	1850.00
4	94	120.00	0.00	40.00	0.00	1.00	0.00	40.00	0.00	0.00	1850.00
4	95	265.00	0.00	20.00	0.00	0.00	0.00	40.00	0.00	0.00	1850.00

4	96	265.00	0.00	20.00	0.00	1.00	0.00	40.00	0.00	0.00	1850.00
5	35	370.00	0.00	10.00	450.00	0.00	10.00	60.00	0.00	0.00	50.00
5	36	370.00	0.00	10.00	370.00	-14.00	10.00	60.00	0.00	0.00	50.00
5	37	370.00	0.00	10.00	370.00	14.00	10.00	60.00	0.00	0.00	50.00
5	38	370.00	-14.00	10.00	370.00	-14.00	20.00	60.00	0.00	0.00	50.00
5	39	350.00	14.00	10.00	370.00	14.00	10.00	60.00	0.00	0.00	50.00
5	40	350.00	-14.00	20.00	370.00	-14.00	20.00	60.00	0.00	0.00	50.00
5	41	320.00	14.00	10.00	350.00	14.00	10.00	60.00	0.00	0.00	50.00
5	42	320.00	-14.00	20.00	350.00	-14.00	20.00	60.00	0.00	0.00	50.00
5	43	290.00	14.00	10.00	320.00	14.00	10.00	60.00	0.00	0.00	50.00
5	44	280.00	14.00	10.00	290.00	14.00	10.00	60.00	0.00	0.00	50.00
5	45	280.00	0.00	10.00	280.00	14.00	10.00	60.00	0.00	0.00	50.00
5	46	280.00	0.00	10.00	290.00	0.00	10.00	60.00	0.00	0.00	50.00
5	47	280.00	0.00	10.00	280.00	-14.00	10.00	60.00	0.00	0.00	50.00
5	48	280.00	-14.00	10.00	280.00	-14.00	20.00	60.00	0.00	0.00	50.00
5	49	290.00	-14.00	20.00	320.00	-14.00	20.00	60.00	0.00	0.00	50.00
5	50	280.00	-14.00	20.00	290.00	-14.00	20.00	60.00	0.00	0.00	50.00
5	51	145.00	0.00	10.00	165.00	0.00	10.00	60.00	0.00	0.00	50.00
5	52	145.00	0.00	10.00	145.00	14.00	10.00	60.00	0.00	0.00	50.00
5	53	145.00	0.00	10.00	145.00	-14.00	10.00	60.00	0.00	0.00	50.00
5	54	145.00	-14.00	10.00	145.00	-14.00	20.00	60.00	0.00	0.00	50.00
5	55	265.00	-14.00	20.00	280.00	-14.00	20.00	60.00	0.00	0.00	50.00
5	56	235.00	-14.00	20.00	265.00	-14.00	20.00	60.00	0.00	0.00	50.00
5	57	200.00	-14.00	20.00	235.00	-14.00	20.00	60.00	0.00	0.00	50.00
5	58	165.00	-14.00	20.00	200.00	-14.00	20.00	60.00	0.00	0.00	50.00
5	59	145.00	-14.00	20.00	165.00	-14.00	20.00	60.00	0.00	0.00	50.00
5	60	135.00	-14.00	20.00	145.00	-14.00	20.00	60.00	0.00	0.00	50.00
5	61	120.00	-14.00	20.00	135.00	-14.00	20.00	60.00	0.00	0.00	50.00
5	62	120.00	-14.00	20.00	120.00	-14.00	40.00	60.00	0.00	0.00	50.00
5	63	120.00	-14.00	40.00	120.00	-10.00	40.00	60.00	0.00	0.00	50.00
5	64	135.00	14.00	10.00	145.00	14.00	10.00	60.00	0.00	0.00	50.00
5	65	145.00	14.00	10.00	135.00	14.00	10.00	60.00	0.00	0.00	50.00
5	66	120.00	14.00	10.00	120.00	14.00	30.00	60.00	0.00	0.00	50.00
5	67	120.00	14.00	30.00	120.00	10.00	30.00	60.00	0.00	0.00	50.00
5	68	265.00	14.00	10.00	280.00	14.00	10.00	60.00	0.00	0.00	50.00
5	69	235.00	14.00	10.00	265.00	14.00	10.00	60.00	0.00	0.00	50.00
5	70	200.00	14.00	10.00	235.00	14.00	10.00	60.00	0.00	0.00	50.00
5	71	165.00	14.00	10.00	200.00	14.00	10.00	60.00	0.00	0.00	50.00
5	72	145.00	14.00	10.00	165.00	14.00	10.00	60.00	0.00	0.00	50.00
5	73	120.00	-10.00	40.00	120.00	0.00	40.00	60.00	0.00	0.00	50.00
5	74	120.00	0.00	40.00	120.00	0.00	30.00	60.00	0.00	0.00	50.00
5	75	120.00	0.00	30.00	120.00	10.00	30.00	60.00	0.00	0.00	50.00
5	76	120.00	10.00	30.00	105.00	10.00	30.00	60.00	0.00	0.00	50.00
5	77	105.00	10.00	30.00	100.00	10.00	30.00	60.00	0.00	0.00	50.00
5	78	105.00	0.00	40.00	120.00	0.00	40.00	60.00	0.00	0.00	50.00
5	79	100.00	0.00	40.00	105.00	0.00	40.00	60.00	0.00	0.00	50.00
5	80	105.00	-10.00	40.00	120.00	-10.00	40.00	60.00	0.00	0.00	50.00
5	81	100.00	-10.00	40.00	105.00	-10.00	40.00	60.00	0.00	0.00	50.00
5	82	265.00	-10.00	20.00	265.00	-14.00	20.00	60.00	0.00	0.00	50.00
5	83	265.00	10.00	10.00	265.00	14.00	10.00	60.00	0.00	0.00	50.00
5	84	265.00	0.00	20.00	265.00	-10.00	20.00	60.00	0.00	0.00	50.00
5	85	265.00	0.00	20.00	265.00	0.00	10.00	60.00	0.00	0.00	50.00
5	86	265.00	0.00	10.00	265.00	10.00	10.00	60.00	0.00	0.00	50.00
5	87	265.00	10.00	10.00	260.00	10.00	10.00	60.00	0.00	0.00	50.00
5	88	255.00	10.00	10.00	260.00	10.00	10.00	60.00	0.00	0.00	50.00
5	89	260.00	0.00	20.00	265.00	0.00	20.00	60.00	0.00	0.00	50.00
5	90	255.00	0.00	20.00	260.00	0.00	20.00	60.00	0.00	0.00	50.00
5	91	260.00	-10.00	20.00	265.00	-10.00	20.00	60.00	0.00	0.00	50.00
5	92	255.00	-10.00	20.00	260.00	-10.00	20.00	60.00	0.00	0.00	50.00

9999

260.00 20.00 25.00 28.00 38.00 180.00 0.05

Appendix B. Output File

```
=======================================================
==       BEAVER   WITH SURVIVABILITY ANALYSIS        ==
==                  MAY 7, 1992                      ==
==       CLIFF WHITCOMB  (FOR MIT THESIS) WITH       ==
==          CHARLES STARK DRAPER LABORATORY          ==
==             SYSTEM SCIENCES DIVISION              ==
==          CAMBRIDGE, MASSACHUSETTS 02139           ==
=======================================================

*****    BEAVER TALE WAS CREATED ON 05/07/92 AT 23:22:28    *****
1

///////////////////////////////////////////////////////////////////////////////
        RUN ID: ELECTRICAL DISTRIBUTION SYSTEM SURVIVABILITY 5/7/92

    NO. OF MISSIONS:                                     5
    MISSIONS BETWEEN SHORT PRINTS:                       5
    LOWEST GROUP NUMBER:                               100
    NO. OF STD. DEVIATIONS:                         1.9600
    NO. OF PHASE TYPES:                                  1
    SEED NO. 1 :                                 113524955
    SEED NO. 2 :                                1332347639
    SEED NO. 3 :                                1332347639
    SEED NO. 4 :                                 113524955

        MISSION TIME LINE
        PHASE NO.        TYPE       DURATION       CUM. TIME
           1               1          100.00          100.00

         REPORT OPTION SELECTED :      4


        SERVER STATUS CODE: 0=OPER. 1=NOT OPER.

        PHASE TYPE                SERVER STATUS
               1                       1

        NO. STAGES OF SERVICE:                  1
///////////////////////////////////////////////////////////////////////////////
         EQUIPMENT TYPES AND PARAMETERS
TYPE        NAME          MTTF      MTTR       DUTYCYCLE    REP. PRIORITY
   1 POWER SOURCE        25000.     6.00         1.00              1
   2 BREAKER              5000.     1.00         1.00              1
   3 LOAD CENTER         10000.     1.00         1.00              1
   4 BUS TIE              5000.     2.00         1.00              1
   5 FCS LOAD             3000.     2.00         1.00
   6 AUX LOAD             1000.     4.00         1.00
   7 ABT                  3000.     3.00         1.00
   8 CABLE                8000.     2.00         1.00              1
///////////////////////////////////////////////////////////////////////////////

ELECTRICAL DISTRIBUTION SYSTEM SURVIVABILITY 5/7/92

EQUIP-TYPE EQUIP-NUMBER MAP
TYPE        NAME               EQUIPMENT NUMBERS OF THIS TYPE
   1   POWER SOURCE       1   2   3   0   0   0   0   0   0   0
   2   BREAKER            4   5   6   7   8   9  10  11  12  13
   2   BREAKER           14  15  16  17  18  93  94  95  96   0
   3   LOAD CENTER       19  20  21  22   0   0   0   0   0   0
   4   BUS TIE           23  24  25  26   0   0   0   0   0   0
   5   FCS LOAD          27  28  29   0   0   0   0   0   0   0
   6   AUX LOAD          30  31  32   0   0   0   0   0   0   0
```

7	ABT	33	34	0	0	0	0	0	0	0	0
8	CABLE	35	36	37	38	39	40	41	42	43	44
8	CABLE	45	46	47	48	49	50	51	52	53	54
8	CABLE	55	56	57	58	59	60	61	62	63	64
8	CABLE	65	66	67	68	69	70	71	72	73	74
8	CABLE	75	76	77	78	79	80	81	82	83	84
8	CABLE	85	86	87	88	89	90	91	92	0	0

UNLIMITED SPARES

//

CONFIGURATION OF SYSTEM

SYSTEM NAME: SYST
SYSTEM GROUP NO: 155

//

GROUP CONFIGURATIONS

GROUP:	100	REQUIRES	3	OF MEMBERS:	4	35	1	0	0	0	0	0	0	0
GROUP:	101	REQUIRES	3	OF MEMBERS:	7	46	2	0	0	0	0	0	0	0
GROUP:	102	REQUIRES	3	OF MEMBERS:	10	51	3	0	0	[illegible]	0	0	0	0
GROUP:	103	REQUIRES	3	OF MEMBERS:	39	37	6	0	0	0	[illegible]	0	0	0
GROUP:	104	REQUIRES	4	OF MEMBERS:	40	38	36	5	0	0	0	0	0	0
GROUP:	105	REQUIRES	2	OF MEMBERS:	45	9	0	0	0	0	0	0	0	0
GROUP:	106	REQUIRES	3	OF MEMBERS:	48	47	8	0	0	0	0	0	0	0
GROUP:	107	REQUIRES	2	OF MEMBERS:	52	12	0	0	0	0	0	0	0	0
GROUP:	108	REQUIRES	3	OF MEMBERS:	54	53	11	0	0	0	0	0	0	0
GROUP:	109	REQUIRES	2	OF MEMBERS:	103	100	0	0	0	0	0	0	0	0
GROUP:	110	REQUIRES	2	OF MEMBERS:	104	100	0	0	0	0	0	0	0	0
GROUP:	111	REQUIRES	2	OF MEMBERS:	105	101	0	0	0	0	0	0	0	0
GROUP:	112	REQUIRES	2	OF MEMBERS:	106	101	0	0	0	0	0	0	0	0
GROUP:	113	REQUIRES	2	OF MEMBERS:	107	102	0	0	0	0	0	0	0	0
GROUP:	114	REQUIRES	2	OF MEMBERS:	108	102	0	0	0	0	0	0	0	0
GROUP:	115	REQUIRES	4	OF MEMBERS:	44	24	43	41	0	0	0	0	0	0
GROUP:	116	REQUIRES	4	OF MEMBERS:	50	23	49	42	0	0	0	0	0	0
GROUP:	117	REQUIRES	2	OF MEMBERS:	115	109	0	0	[illegible]	0	0	0	0	0
GROUP:	118	REQUIRES	2	OF MEMBERS:	116	110	0	0	0	0	0	0	0	0
GROUP:	119	REQUIRES	1	OF MEMBERS:	111	117	0	0	0	0	0	0	0	0
GROUP:	120	REQUIRES	1	OF MEMBERS:	112	118	0	0	0	0	0	0	0	0
GROUP:	121	REQUIRES	2	OF MEMBERS:	68	119	0	0	0	0	0	0	0	0
GROUP:	122	REQUIRES	2	OF MEMBERS:	55	120	0	0	0	0	0	0	0	0
GROUP:	123	REQUIRES	5	OF MEMBERS:	59	25	58	57	56	0	0	0	0	0
GROUP:	124	REQUIRES	5	OF MEMBERS:	72	26	71	70	69	0	0	0	0	0
GROUP:	125	REQUIRES	2	OF MEMBERS:	124	113	0	0	0	0	0	0	0	0
GROUP:	126	REQUIRES	2	OF MEMBERS:	123	114	0	0	0	0	0	0	0	0
GROUP:	127	REQUIRES	1	OF MEMBERS:	122	126	0	0	0	0	0	0	0	0
GROUP:	128	REQUIRES	1	OF MEMBERS:	121	125	0	0	0	0	0	0	0	0
GROUP:	129	REQUIRES	2	OF MEMBERS:	82	127	0	0	0	0	0	0	0	0
GROUP:	130	REQUIRES	2	OF MEMBERS:	83	128	0	0	0	0	0	0	0	0
GROUP:	131	REQUIRES	6	OF MEMBERS:	30	92	16	91	21	129	0	0	0	0
GROUP:	132	REQUIRES	6	OF MEMBERS:	31	88	18	87	22	130	0	0	0	0
GROUP:	133	REQUIRES	4	OF MEMBERS:	95	84	21	129	0	0	0	0	0	0
GROUP:	134	REQUIRES	5	OF MEMBERS:	96	85	86	22	130	0	0	0	0	0
GROUP:	135	REQUIRES	1	OF MEMBERS:	133	134	0	0	0	0	0	0	0	0
GROUP:	136	REQUIRES	2	OF MEMBERS:	34	135	0	0	0	0	0	0	0	0
GROUP:	137	REQUIRES	5	OF MEMBERS:	32	90	17	89	136	0	0	0	0	0
GROUP:	138	REQUIRES	2	OF MEMBERS:	123	122	0	0	0	0	0	0	0	0
GROUP:	139	REQUIRES	1	OF MEMBERS:	114	138	0	0	0	0	0	0	0	0
GROUP:	140	REQUIRES	2	OF MEMBERS:	60	139	0	0	0	0	0	0	0	0
GROUP:	141	REQUIRES	2	OF MEMBERS:	124	121	0	0	0	0	0	0	0	0

GROUP:	142 REQUIRES	1 OF MEMBERS:	113	141	0	0	0	0	0	0	0	0
GROUP:	143 REQUIRES	2 OF MEMBERS:	64	142	0	0	0	0	0	0	0	0
GROUP:	144 REQUIRES	5 OF MEMBERS:	19	63	62	61	140	0	0	0	0	0
GROUP:	145 REQUIRES	5 OF MEMBERS:	27	81	13	80	144	0	0	0	0	0
GROUP:	146 REQUIRES	5 OF MEMBERS:	20	67	66	65	143	0	0	0	0	0
GROUP:	147 REQUIRES	5 OF MEMBERS:	28	77	15	76	146	0	0	0	0	0
GROUP:	148 REQUIRES	3 OF MEMBERS:	93	73	144	0	0	0	0	0	0	0
GROUP:	149 REQUIRES	4 OF MEMBERS:	94	74	75	146	0	0	0	0	0	0
GROUP:	150 REQUIRES	1 OF MEMBERS:	148	149	0	0	0	0	0	0	0	0
GROUP:	151 REQUIRES	2 OF MEMBERS:	33	150	0	0	0	0	0	0	0	0
GROUP:	152 REQUIRES	5 OF MEMBERS:	29	79	14	78	151	0	0	0	0	0
GROUP:	153 REQUIRES	1 OF MEMBERS:	131	132	0	0	0	0	0	0	0	0
GROUP:	154 REQUIRES	1 OF MEMBERS:	145	147	0	0	0	0	0	0	0	0
GROUP:	155 REQUIRES	4 OF MEMBERS:	153	154	137	152	0	0	0	0	0	0

EQDATA I DATA FIELDS 1 - 10

I										
1	450.00	.00	10.00	1800.00	.00	.00	50.00	4.00	.00	5550.03
2	290.00	.00	10.00	1800.00	.00	.00	50.00	7.00	.00	5550.00
3	165.00	.00	10.00	1800.00	.00	.00	50.00	10.00	.00	5550.00
4	370.00	.00	10.00	.00	.00	.00	30.00	5.00	6.00	600.00
5	370.00	.00	10.00	.00	.00	.00	30.00	.00	.00	430.00
6	370.00	.00	10.00	.00	.00	.00	30.00	.00	.00	430.00
7	280.00	.00	10.00	.00	.00	.00	30.00	8.00	9.00	500.00
8	280.00	.00	10.00	.00	.00	.00	30.00	.00	.00	430.00
9	280.00	.00	10.00	.00	.00	.00	30.00	.00	.00	430.00
10	145.00	.00	10.00	.00	.00	.00	30.00	11.00	12.00	500.00
11	145.00	.00	10.00	.00	.00	.00	30.00	.00	.00	430.00
12	145.00	.00	10.00	.00	.00	.00	30.00	.00	.00	430.00
13	105.00	-10.00	40.00	.00	.00	.00	30.00	.00	.00	630.00
14	105.00	10.00	40.00	.00	.00	.00	30.00	.00	.00	630.00
15	105.00	.00	30.00	.00	.00	.00	30.00	.00	.00	630.00
16	260.00	-10.00	20.00	.00	.00	.00	30.00	.00	.00	630.00
17	260.00	.00	20.00	.00	.00	.00	30.00	.00	.00	630.00
18	260.00	10.00	10.00	.00	.00	.00	30.00	.00	.00	630.00
19	120.00	-10.00	40.00	.00	.00	.00	90.00	.00	.00	1250.00
20	120.00	10.00	30.00	.00	.00	.00	90.00	.00	.00	1250.00
21	265.00	-10.00	20.00	.00	.00	.00	90.00	.00	.00	1250.00
22	265.00	10.00	10.00	.00	.00	.00	90.00	.00	.00	1250.00
23	290.00	-14.00	20.00	.00	.00	.00	50.00	.00	.00	3250.00
24	290.00	14.00	10.00	.00	.00	.00	50.00	.00	.00	3250.00
25	165.00	-14.00	20.00	.00	1.00	.00	50.00	.00	.00	3250.00
26	165.00	14.00	10.00	.00	1.00	.00	50.00	.00	.00	3250.00
27	100.00	-10.00	40.00	600.00	.00	.00	50.00	13.00	.00	8900.00
28	100.00	10.00	30.00	1400.00	.00	.00	50.00	15.00	.00	7900.00
29	100.00	.00	40.00	400.00	.00	.00	80.00	14.00	.00	6700.00
30	255.00	-10.00	20.00	650.00	.00	.00	60.00	16.00	.00	5600.00
31	255.00	10.00	10.00	850.00	.00	.00	60.00	18.00	.00	3900.00
32	255.00	.00	20.00	400.00	.00	.00	80.00	17.00	.00	9800.00
33	120.00	.00	40.00	.00	.00	.00	30.00	93.00	94.00	4000.00
34	265.00	.00	20.00	.00	.00	.00	30.00	95.00	96.00	4000.00
35	370.00	.00	10.00	450.00	.00	10.00	60.00	.00	.00	50.00
36	370.00	.00	10.00	370.00	-14.00	10.00	60.00	.00	.00	50.00
37	370.00	.00	10.00	370.00	14.00	10.00	60.00	.00	.00	50.00
38	370.00	-14.00	10.00	370.00	-14.00	20.00	60.00	.00	.00	50.00
39	350.00	14.00	10.00	370.00	14.00	10.00	60.00	.00	.00	50.00
40	350.00	-14.00	20.00	370.00	-14.00	20.00	60.00	.00	.00	50.00
41	320.00	14.00	10.00	350.00	14.00	10.00	60.00	.00	.00	50.00
42	320.00	-14.00	20.00	350.00	-14.00	20.00	60.00	.00	.00	50.00
43	290.00	14.00	10.00	320.00	14.00	10.00	60.00	.00	.00	50.00
44	280.00	14.00	10.00	290.00	14.00	10.00	60.00	.00	.00	50.00
45	280.00	.00	10.00	280.00	14.00	10.00	60.00	.00	.00	50.00
46	280.00	.00	10.00	290.00	.00	10.00	60.00	.00	.00	50.00
47	280.00	.00	10.00	280.00	-14.00	10.00	60.00	.00	.00	50.00

48	280.00	-14.00	10.00	280.00	-14.00	20.00	60.00	.00	.00	50.00
49	290.00	-14.00	20.00	320.00	-14.00	20.00	60.00	.00	.00	50.00
50	280.00	-14.00	20.00	290.00	-14.00	20.00	60.00	.00	.00	50.00
51	145.00	.00	10.00	165.00	.00	10.00	60.00	.00	.00	50.00
52	145.00	.00	10.00	145.00	14.00	10.00	60.00	.00	.00	50.00
53	145.00	.00	10.00	145.00	-14.00	10.00	60.00	.00	.00	50.00
54	145.00	-14.00	10.00	145.00	-14.00	20.00	60.00	.00	.00	50.00
55	265.00	-14.00	20.00	280.00	-14.00	20.00	60.00	.00	.00	50.00
56	235.00	-14.00	20.00	265.00	-14.00	20.00	60.00	.00	.00	50.00
57	200.00	-14.00	20.00	235.00	-14.00	20.00	60.00	.00	.00	50.00
58	165.00	-14.00	20.00	200.00	-14.00	20.00	60.00	.00	.00	50.00
59	145.00	-14.00	20.00	165.00	-14.00	20.00	60.00	.00	.00	50.00
60	135.00	-14.00	20.00	145.00	-14.00	20.00	60.00	.00	.00	50.00
61	120.00	-14.00	20.00	135.00	-14.00	20.00	60.00	.00	.00	50.00
62	120.00	-14.00	20.00	120.00	-14.00	40.00	60.00	.00	.00	50.00
63	120.00	-14.00	40.00	120.00	-10.00	40.00	60.00	.00	.00	50.00
64	135.00	14.00	10.00	145.00	14.00	10.00	60.00	.00	.00	50.00
65	145.00	14.00	10.00	135.00	14.00	10.00	60.00	.00	.00	50.00
66	120.00	14.00	10.00	120.00	14.00	30.00	60.00	.00	.00	50.00
67	120.00	14.00	30.00	120.00	10.00	30.00	60.00	.00	.00	50.00
68	265.00	14.00	10.00	280.00	14.00	10.00	60.00	.00	.00	50.00
69	235.00	14.00	10.00	265.00	14.00	10.00	60.00	.00	.00	50.00
70	200.00	14.00	10.00	235.00	14.00	10.00	60.00	.00	.00	50.00
71	165.00	14.00	10.00	200.00	14.00	10.00	60.00	.00	.00	50.00
72	145.00	14.00	10.00	165.00	14.00	10.00	60.00	.00	.00	50.00
73	120.00	-10.00	40.00	120.00	.00	40.00	60.00	.00	.00	50.00
74	120.00	.00	40.00	120.00	.00	30.00	60.00	.00	.00	50.00
75	120.00	.00	30.00	120.00	10.00	30.00	60.00	.00	.00	50.00
76	120.00	10.00	30.00	105.00	10.00	30.00	60.00	.00	.00	50.00
77	105.00	10.00	30.00	100.00	10.00	30.00	60.00	.00	.00	50.00
78	105.00	.00	40.00	120.00	.00	40.00	60.00	.00	.00	50.00
79	100.00	.00	40.00	105.00	.00	40.00	60.00	.00	.00	50.00
80	105.00	-10.00	40.00	120.00	-10.00	40.00	60.00	.00	.00	50.00
81	100.00	-10.00	40.00	105.00	-10.00	40.00	60.00	.00	.00	50.00
82	265.00	-10.00	20.00	265.00	-14.00	20.00	60.00	.00	.00	50.00
83	265.00	10.00	10.00	265.00	14.00	10.00	60.00	.00	.00	50.00
84	265.00	.00	20.00	265.00	-10.00	20.00	60.00	.00	.00	50.00
85	265.00	.00	20.00	265.00	.00	10.00	60.00	.00	.00	50.00
86	265.00	.00	10.00	265.00	10.00	10.00	60.00	.00	.00	50.00
87	265.00	10.00	10.00	260.00	10.00	10.00	60.00	.00	.00	50.00
88	255.00	10.00	10.00	260.00	10.00	10.00	60.00	.00	.00	50.00
89	260.00	.00	20.00	265.00	.00	20.00	60.00	.00	.00	50.00
90	255.00	.00	20.00	260.00	.00	20.00	60.00	.00	.00	50.00
91	260.00	-10.00	20.00	265.00	-10.00	20.00	60.00	.00	.00	50.00
92	255.00	-10.00	20.00	260.00	-10.00	20.00	60.00	.00	.00	50.00
93	120.00	.00	40.00	.00	.00	.00	40.00	.00	.00	1850.00
94	120.00	.00	40.00	.00	1.00	.00	40.00	.00	.00	1850.00
95	265.00	.00	20.00	.00	.00	.00	40.00	.00	.00	1850.00
96	265.00	.00	20.00	.00	1.00	.00	40.00	.00	.00	1850.00

TRANSPOSE OF POWER PATH CONTINUITY ARRAY

27	27	27	28	28	28	29	29	29	29	29	29	30	30	30	31	31	31	32	32	32	32	32	32
81	81	81	77	77	77	79	79	79	79	79	79	92	92	92	88	88	88	90	90	90	90	90	90
13	13	13	15	15	15	14	14	14	14	14	14	16	16	16	18	18	18	17	17	17	17	17	17
80	80	80	76	76	76	78	78	78	78	78	78	91	91	91	87	87	87	89	89	89	89	89	89
19	19	19	20	20	20	33	33	33	33	33	33	21	21	21	22	22	22	34	34	34	34	34	34
63	63	63	67	67	67	93	93	93	94	94	94	82	82	82	83	83	83	95	95	95	96	96	96
62	62	62	66	66	66	73	73	73	74	74	74	55	55	59	68	68	72	84	84	84	85	85	85
61	61	61	65	65	65	19	19	19	75	75	75	48	50	25	45	44	26	21	21	21	86	86	86
60	60	60	64	64	64	63	63	63	20	20	20	47	23	58	9	24	71	82	82	82	22	22	22
54	59	59	52	72	72	62	62	62	67	67	67	8	49	57	7	43	70	55	55	59	83	83	83
53	25	25	12	26	26	61	61	61	66	66	66	7	42	56	46	41	69	48	50	25	68	68	72
11	58	58	10	71	71	60	60	60	65	65	65	46	40	54	2	39	52	47	23	58	45	44	26
10	57	57	51	70	70	54	59	59	64	64	64	2	38	53	0	37	12	8	49	57	9	24	71
51	56	56	3	69	69	53	25	25	52	72	72	0	36	11	0	6	10	7	42	56	7	43	70
3	55	55	0	68	68	11	58	58	12	26	26	0	5	10	0	4	51	46	40	54	46	41	69
0	48	50	0	45	44	10	57	57	10	71	71	0	4	51	0	35	3	2	38	53	2	39	52
0	47	23	0	9	24	51	56	56	51	70	70	0	35	3	0	1	0	0	36	11	0	37	12
0	8	49	0	7	43	3	55	55	3	69	69	0	1	0	0	0	0	0	5	10	0	6	10
0	7	42	0	46	41	0	48	50	0	68	68	0	0	0	0	0	0	0	4	51	0	4	51
0	46	40	0	2	39	0	47	23	0	45	44	0	0	0	0	0	0	0	35	3	0	35	3
0	2	38	0	0	37	0	8	49	0	9	24	0	0	0	0	0	0	0	1	0	0	1	0
0	0	36	0	0	6	0	7	42	0	7	43	0	0	0	0	0	0	0	0	0	0	0	0
0	0	5	0	0	4	0	46	40	0	46	41	0	0	0	0	0	0	0	0	0	0	0	0
0	0	4	0	0	35	0	2	38	0	2	39	0	0	0	0	0	0	0	0	0	0	0	0
0	0	35	0	0	1	0	0	36	0	0	37	0	0	0	0	0	0	0	0	0	0	0	0
0	0	1	0	0	0	0	0	5	0	0	6	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	4	0	0	4	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	35	0	0	35	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	1	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0

TRANSPOSE OF SOURCE PATH CONTINUITY ARRAY

3	2	1	3	2	1	3	2	1	3	2	1	2	1	3	2	1	3	2	1	3	2	1	3
51	46	35	51	46	35	51	46	35	51	46	35	46	35	51	46	35	51	46	35	51	46	35	51
10	7	4	10	7	4	10	7	4	10	7	4	7	4	10	7	4	10	7	4	10	7	4	10
11	8	5	12	9	6	11	8	5	12	9	6	8	5	11	9	6	12	8	5	11	9	6	12
53	47	36	52	45	37	53	47	36	52	45	37	47	36	53	45	37	52	47	36	53	45	37	52
54	48	38	64	68	39	54	48	38	64	68	39	48	38	54	68	39	69	48	38	54	68	39	69
60	55	40	65	69	41	60	55	40	65	69	41	55	40	56	83	41	70	55	40	56	83	41	70
61	56	42	66	70	43	61	56	42	66	70	43	82	42	57	22	43	71	82	42	57	22	43	71
62	57	49	67	71	24	62	57	49	67	71	24	21	49	58	87	24	26	21	49	58	86	24	26
63	58	23	20	26	44	63	58	23	20	26	44	91	23	25	18	44	72	84	23	25	85	44	72
19	25	50	76	72	68	19	25	50	75	72	68	16	50	59	88	68	83	95	50	59	96	68	83
80	59	55	15	64	69	73	59	55	74	64	69	92	55	82	31	83	22	34	55	82	34	83	22
13	60	56	77	65	70	93	60	56	94	65	70	30	82	21	0	22	87	89	82	21	89	22	86
81	61	57	28	66	71	33	61	57	33	66	71	0	21	91	0	87	18	17	21	84	17	86	85
27	62	58	0	67	26	78	62	58	78	67	26	0	91	16	0	18	88	90	84	95	90	85	96
0	63	25	0	20	72	14	63	25	14	20	72	0	16	92	0	88	31	32	95	34	32	96	34
0	19	59	0	76	64	79	19	59	79	75	64	0	92	30	0	31	0	0	34	89	0	34	89
0	80	60	0	15	65	29	73	60	29	74	65	0	30	0	0	0	0	0	89	17	0	89	17
0	13	61	0	77	66	0	93	61	0	94	66	0	0	0	0	0	0	0	17	90	0	17	90
0	81	62	0	28	67	0	33	62	0	33	67	0	0	0	0	0	0	0	90	32	0	90	32
0	27	63	0	0	20	0	78	63	0	78	20	0	0	0	0	0	0	0	32	0	0	32	0
0	0	19	0	0	76	0	14	19	0	14	75	0	0	0	0	0	0	0	0	0	0	0	0
0	0	80	0	0	15	0	79	73	0	79	74	0	0	0	0	0	0	0	0	0	0	0	0
0	0	13	0	0	77	0	29	93	0	29	94	0	0	0	0	0	0	0	0	0	0	0	0
0	0	81	0	0	28	0	0	33	0	0	33	0	0	0	0	0	0	0	0	0	0	0	0
0	0	27	0	0	0	0	0	78	0	0	78	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	14	0	0	14	0	0	[illegible]	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	79	0	0	79	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	29	0	0	29	0	0	0	0	0	0	0	0	0	0	0	0

HIT LOCATION IS 260.00 20.00 25.00

```
I =    1 RADIUS =     191.64
I =    2 RADIUS =      39.05
I =    3 RADIUS =      98.23
I =    4 RADIUS =     112.81
I =    5 RADIUS =     112.81
I =    6 RADIUS =     112.81
I =    7 RADIUS =      32.02
I =    8 RADIUS =      32.02
I =    9 RADIUS =      32.02
I =   10 RADIUS =     117.69
I =   11 RADIUS =     117.69
I =   12 RADIUS =     117.69
I =   13 RADIUS =     158.59
I =   14 RADIUS =     156.04
I =   15 RADIUS =     156.36
I =   16 RADIUS =      30.41
I =   17 RADIUS =      20.62
I =   18 RADIUS =      18.03
I =   19 RADIUS =     143.96
I =   20 RADIUS =     140.45
I =   21 RADIUS =      30.82
I =   22 RADIUS =      18.71
I =   23 RADIUS =      45.62
I =   24 RADIUS =      34.07
I =   25 RADIUS =     101.02
I =   26 RADIUS =      96.36
I =   27 RADIUS =     163.48
I =   28 RADIUS =     160.39
I =   29 RADIUS =     161.94
I =   30 RADIUS =      30.82
I =   31 RADIUS =      18.71
I =   32 RADIUS =      21.21
I =   33 RADIUS =     142.21
I =   34 RADIUS =      21.21
I =   35 RADIUS =     112.81
I =   36 RADIUS =     112.81
I =   37 RADIUS =     111.18
I =   38 RADIUS =     115.24
I =   39 RADIUS =      91.44
I =   40 RADIUS =      96.34
I =   41 RADIUS =      62.14
I =   42 RADIUS =      69.14
I =   43 RADIUS =      34.07
I =   44 RADIUS =      25.71
I =   45 RADIUS =      25.71
I =   46 RADIUS =      32.02
I =   47 RADIUS =      32.02
I =   48 RADIUS =      39.76
I =   49 RADIUS =      45.62
I =   50 RADIUS =      39.76
I =   51 RADIUS =      98.23
I =   52 RADIUS =     116.13
I =   53 RADIUS =     117.69
I =   54 RADIUS =     120.02
I =   55 RADIUS =      34.73
I =   56 RADIUS =      34.37
I =   57 RADIUS =      42.50
I =   58 RADIUS =      69.14
I =   59 RADIUS =     101.02
I =   60 RADIUS =     120.02
I =   61 RADIUS =     129.64
I =   62 RADIUS =     144.07
```

```
I =   63 RADIUS =     143.96
I =   64 RADIUS =     116.13
I =   65 RADIUS =     116.13
I =   66 RADIUS =     140.13
I =   67 RADIUS =     140.22
I =   68 RADIUS =      16.91
I =   69 RADIUS =      16.16
I =   70 RADIUS =      29.77
I =   71 RADIUS =      62.14
I =   72 RADIUS =      96.36
I =   73 RADIUS =     142.21
I =   74 RADIUS =     141.51
I =   75 RADIUS =     140.45
I =   76 RADIUS =     140.45
I =   77 RADIUS =     155.40
I =   78 RADIUS =     142.21
I =   79 RADIUS =     157.00
I =   80 RADIUS =     143.96
I =   81 RADIUS =     158.59
I =   82 RADIUS =      30.82
I =   83 RADIUS =      16.91
I =   84 RADIUS =      21.21
I =   85 RADIUS =      21.21
I =   86 RADIUS =      18.71
I =   87 RADIUS =      18.03
I =   88 RADIUS =      18.03
I =   89 RADIUS =      20.62
I =   90 RADIUS =      20.62
I =   91 RADIUS =      30.41
I =   92 RADIUS =      30.41
I =   93 RADIUS =     142.21
I =   94 RADIUS =     142.21
I =   95 RADIUS =      21.21
I =   96 RADIUS =      21.21

SWITCHING EQUIPMENT #   25 OFF BEFORE ANALYSIS.


SWITCHING EQUIPMENT #   26 OFF BEFORE ANALYSIS.


SWITCHING EQUIPMENT #   94 OFF BEFORE ANALYSIS.


SWITCHING EQUIPMENT #   96 OFF BEFORE ANALYSIS.


THE DAMAGE RADIUS IS    32.45 FEET.

LOOKING FOR ALL SOURCES FOR LOAD   27

FOUND SOURCE #    3

LOOKING FOR ALL SOURCES FOR LOAD   28

FOUND SOURCE #    3

LOOKING FOR ALL SOURCES FOR LOAD   29

FOUND SOURCE #    3

LOOKING FOR ALL SOURCES FOR LOAD   30
LOOKING FOR ALL SOURCES FOR LOAD   31
LOOKING FOR ALL SOURCES FOR LOAD   32
```

```
GENERATOR/LOAD MATRIX: ROW = SOURCE #, COL = LOAD #

     .0     .0     .0     .0     .0     .0
     .0     .0     .0     .0     .0     .0
    1.0    1.0    1.0     .0     .0     .0

PLLCHK(   1) =      1.00

PLLCHK(   2) =      1.00

PLLCHK(   3) =      1.00

PLLCHK(   4) =       .00

PLLCHK(   5) =       .00

PLLCHK(   6) =       .00

LOAD    4, EQUIPMENT #    30 NOT POWERED.

LOAD    5, EQUIPMENT #    31 DAMAGED.

LOAD    6, EQUIPMENT #    32 NOT POWERED.

SRCLD(   1) =       .00

SRCLD(   2) =       .00

SOURCE(   3) IS OVERLOADED BY   600.00 KW.

SRCLD(   3) =  2400.00

LOAD   30 NOT POWERED, BUT IT SHOULD BE.

LOAD   32 NOT POWERED, BUT IT SHOULD BE.

SOURCE(   3) IS OVERLOADED AND NEEDS ATTENTION.

SWITCHING EQUIPMENT #   25 OFF BEFORE ANALYSIS.

SWITCHING EQUIPMENT #   26 OFF BEFORE ANALYSIS.

SWITCHING EQUIPMENT #   94 OFF BEFORE ANALYSIS.

SWITCHING EQUIPMENT #   96 OFF BEFORE ANALYSIS.

THE DAMAGE RADIUS IS    32.72 FEET.

LOOKING FOR ALL SOURCES FOR LOAD   27

FOUND SOURCE #     3

LOOKING FOR ALL SOURCES FOR LOAD   28

FOUND SOURCE #     3

LOOKING FOR ALL SOURCES FOR LOAD   29

FOUND SOURCE #     3
```

```
LOOKING FOR ALL SOURCES FOR LOAD   30
LOOKING FOR ALL SOURCES FOR LOAD   31
LOOKING FOR ALL SOURCES FOR LOAD   32

GENERATOR/LOAD MATRIX: ROW = SOURCE #, COL = LOAD #

      .0      .0      .0      .0      .0      .0
      .0      .0      .0      .0      .0      .0
     1.0     1.0     1.0      .0      .0      .0

PLLCHK(   1) =      1.00

PLLCHK(   2) =      1.00

PLLCHK(   3) =      1.00

PLLCHK(   4) =       .00

PLLCHK(   5) =       .00

PLLCHK(   6) =       .00

LOAD    4, EQUIPMENT #   30 NOT POWERED.

LOAD    5, EQUIPMENT #   31 DAMAGED.

LOAD    6, EQUIPMENT #   32 NOT POWERED.

SRCLD(   1) =       .00

SRCLD(   2) =       .00

SOURCE(   3) IS OVERLOADED BY   600.00 KW.

SRCLD(   3) =  2400.00

LOAD   30 NOT POWERED, BUT IT SHOULD BE.

LOAD   32 NOT POWERED, BUT IT SHOULD BE.

SOURCE(   3) IS OVERLOADED AND NEEDS ATTENTION.

SWITCHING EQUIPMENT #   25 OFF BEFORE ANALYSIS.

SWITCHING EQUIPMENT #   26 OFF BEFORE ANALYSIS.

SWITCHING EQUIPMENT #   94 OFF BEFORE ANALYSIS.

SWITCHING EQUIPMENT #   96 OFF BEFORE ANALYSIS.

THE DAMAGE RADIUS IS    28.42 FEET.

LOOKING FOR ALL SOURCES FOR LOAD   27

FOUND SOURCE #     3

LOOKING FOR ALL SOURCES FOR LOAD   28

FOUND SOURCE #     3
```

```
LOOKING FOR ALL SOURCES FOR LOAD   29

FOUND SOURCE #     3

LOOKING FOR ALL SOURCES FOR LOAD   30

FOUND SOURCE #     2


FOUND SOURCE #     1

LOOKING FOR ALL SOURCES FOR LOAD   31
LOOKING FOR ALL SOURCES FOR LOAD   32

GENERATOR/LOAD MATRIX: ROW = SOURCE #, COL = LOAD #

      .0      .0      .0     1.0      .0      .0
      .0      .0      .0     1.0      .0      .0
     1.0     1.0     1.0      .0      .0      .0

PLLCHK(   1)  =      1.00

PLLCHK(   2)  =      1.00

PLLCHK(   3)  =      1.00

PLLCHK(   4)  =      2.00

PLLCHK(   5)  =       .00

PLLCHK(   6)  =       .00

LOAD     5, EQUIPMENT #    31 DAMAGED.

LOAD     6, EQUIPMENT #    32 NOT POWERED.

SRCLD(   1)  =    325.00

SRCLD(   2)  =    325.00

SOURCE(   3) IS OVERLOADED BY    600.00 KW.

SRCLD(   3)  =  2400.00

LOAD    32 NOT POWERED, BUT IT SHOULD BE.

SOURCE(   3) IS OVERLOADED AND NEEDS ATTENTION.

SWITCHING EQUIPMENT #    25 OFF BEFORE ANALYSIS.


SWITCHING EQUIPMENT #    26 OFF BEFORE ANALYSIS.


SWITCHING EQUIPMENT #    94 OFF BEFORE ANALYSIS.


SWITCHING EQUIPMENT #    96 OFF BEFORE ANALYSIS.


THE DAMAGE RADIUS IS     31.95 FEET.

LOOKING FOR ALL SOURCES FOR LOAD   27
```

```
FOUND SOURCE #     3

LOOKING FOR ALL SOURCES FOR LOAD   28

FOUND SOURCE #     3

LOOKING FOR ALL SOURCES FOR LOAD   29

FOUND SOURCE #     3

LOOKING FOR ALL SOURCES FOR LOAD   30
LOOKING FOR ALL SOURCES FOR LOAD   31
LOOKING FOR ALL SOURCES FOR LOAD   32

GENERATOR/LOAD MATRIX: ROW = SOURCE #, COL = LOAD #

      .0      .0      .0      .0      .0      .0
      .0      .0      .0      .0      .0      .0
     1.0     1.0     1.0      .0      .0      .0

PLLCHK(   1) =      1.00

PLLCHK(   2) =      1.00

PLLCHK(   3) =      1.00

PLLCHK(   4) =       .00

PLLCHK(   5) =       .00

PLLCHK(   6) =       .00

LOAD     4, EQUIPMENT #    30 NOT POWERED.

LOAD     5, EQUIPMENT #    31 DAMAGED.

LOAD     6, EQUIPMENT #    32 NOT POWERED.

SRCLD(   1) =       .00

SRCLD(   2) =       .00

SOURCE(   3) IS OVERLOADED BY    600.00 KW.

SRCLD(   3) =  2400.00

LOAD    30 NOT POWERED, BUT IT SHOULD BE.

LOAD    32 NOT POWERED, BUT IT SHOULD BE.

SOURCE(   3) IS OVERLOADED AND NEEDS ATTENTION.

SWITCHING EQUIPMENT #   25 OFF BEFORE ANALYSIS.

SWITCHING EQUIPMENT #   26 OFF BEFORE ANALYSIS.

SWITCHING EQUIPMENT #   94 OFF BEFORE ANALYSIS.

SWITCHING EQUIPMENT #   96 OFF BEFORE ANALYSIS.
```

```
THE DAMAGE RADIUS IS    37.03 FEET.

LOOKING FOR ALL SOURCES FOR LOAD   27

FOUND SOURCE #     3

LOOKING FOR ALL SOURCES FOR LOAD   28

FOUND SOURCE #     3

LOOKING FOR ALL SOURCES FOR LOAD   29

FOUND SOURCE #     3

LOOKING FOR ALL SOURCES FOR LOAD   30
LOOKING FOR ALL SOURCES FOR LOAD   31
LOOKING FOR ALL SOURCES FOR LOAD   32

GENERATOR/LOAD MATRIX: ROW = SOURCE #, COL = LOAD #

      .0      .0      .0      .0      .0      .0
      .0      .0      .0      .0      .0      .0
     1.0     1.0     1.0      .0      .0      .0

PLLCHK(   1) =      1.00

PLLCHK(   2) =      1.00

PLLCHK(   3) =      1.00

PLLCHK(   4) =       .00

PLLCHK(   5) =       .00

PLLCHK(   6) =       .00

LOAD     4, EQUIPMENT #    30 NOT POWERED.

LOAD     5, EQUIPMENT #    31 DAMAGED.

LOAD     6, EQUIPMENT #    32 NOT POWERED.

SRCLD(   1) =       .00

SRCLD(   2) =       .00

SOURCE(   3) IS OVERLOADED BY    600.00 KW.

SRCLD(   3) =  2400.00

LOAD   30 NOT POWERED, BUT IT SHOULD BE.

LOAD   32 NOT POWERED, BUT IT SHOULD BE.

SOURCE(   3) IS OVERLOADED AND NEEDS ATTENTION.
```

THE SURVIVABILITY OF EACH EQUIPMENT (IN PERCENT)
FOR 5 SURVIVABILITY TRIALS.

EQUIPMENT #	1:	100.00 PERCENT.
EQUIPMENT #	2:	100.00 PERCENT.
EQUIPMENT #	3:	100.00 PERCENT.
EQUIPMENT #	4:	100.00 PERCENT.
EQUIPMENT #	5:	100.00 PERCENT.
EQUIPMENT #	6:	100.00 PERCENT.
EQUIPMENT #	7:	40.00 PERCENT.
EQUIPMENT #	8:	40.00 PERCENT.
EQUIPMENT #	9:	40.00 PERCENT.
EQUIPMENT #	10:	100.00 PERCENT.
EQUIPMENT #	11:	100.00 PERCENT.
EQUIPMENT #	12:	100.00 PERCENT.
EQUIPMENT #	13:	100.00 PERCENT.
EQUIPMENT #	14:	100.00 PERCENT.
EQUIPMENT #	15:	100.00 PERCENT.
EQUIPMENT #	16:	20.00 PERCENT.
EQUIPMENT #	17:	.00 PERCENT.
EQUIPMENT #	18:	.00 PERCENT.
EQUIPMENT #	19:	100.00 PERCENT.
EQUIPMENT #	20:	100.00 PERCENT.
EQUIPMENT #	21:	100.00 PERCENT.
EQUIPMENT #	22:	100.00 PERCENT.
EQUIPMENT #	23:	100.00 PERCENT.
EQUIPMENT #	24:	100.00 PERCENT.
EQUIPMENT #	25:	100.00 PERCENT.
EQUIPMENT #	26:	100.00 PERCENT.
EQUIPMENT #	27:	100.00 PERCENT.
EQUIPMENT #	28:	100.00 PERCENT.
EQUIPMENT #	29:	100.00 PERCENT.
EQUIPMENT #	30:	100.00 PERCENT.
EQUIPMENT #	31:	.00 PERCENT.
EQUIPMENT #	32:	100.00 PERCENT.
EQUIPMENT #	33:	100.00 PERCENT.
EQUIPMENT #	34:	.00 PERCENT.
FQUIPMENT #	35:	100.00 PERCENT.
EQUIPMENT #	36:	100.00 PERCENT.
EQUIPMENT #	37:	100.00 PERCENT.
EQUIPMENT #	38:	100.00 PERCENT.
EQUIPMENT #	39:	100.00 PERCENT.
EQUIPMENT #	40:	100.00 PERCENT.
EQUIPMENT #	41:	100.00 PERCENT.
EQUIPMENT #	42:	100.00 PERCENT.
EQUIPMENT #	43:	100.00 PERCENT.
EQUIPMENT #	44:	100.00 PERCENT.
EQUIPMENT #	45:	100.00 PERCENT.
EQUIPMENT #	46:	100.00 PERCENT.
EQUIPMENT #	47:	100.00 PERCENT.
EQUIPMENT #	48:	100.00 PERCENT.
EQUIPMENT #	49:	100.00 PERCENT.
EQUIPMENT #	50:	100.00 PERCENT.
EQUIPMENT #	51:	100.00 PERCENT.
EQUIPMENT #	52:	100.00 PERCENT.
EQUIPMENT #	53:	100.00 PERCENT.
EQUIPMENT #	54:	100.00 PERCENT.
EQUIPMENT #	55:	100.00 PERCENT.
EQUIPMENT #	56:	100.00 PERCENT.
EQUIPMENT #	57:	100.00 PERCENT.
EQUIPMENT #	58:	100.00 PERCENT.
EQUIPMENT #	59:	100.00 PERCENT.
EQUIPMENT #	60:	100.00 PERCENT.
EQUIPMENT #	61:	100.00 PERCENT.

EQUIPMENT #	62:	100.00 PERCENT.
EQUIPMENT #	63:	100.00 PERCENT.
EQUIPMENT #	64:	100.00 PERCENT.
EQUIPMENT #	65:	100.00 PERCENT.
EQUIPMENT #	66:	100.00 PERCENT.
EQUIPMENT #	67:	100.00 PERCENT.
EQUIPMENT #	68:	.00 PERCENT.
EQUIPMENT #	69:	.00 PERCENT.
EQUIPMENT #	70:	100.00 PERCENT.
EQUIPMENT #	71:	100.00 PERCENT.
EQUIPMENT #	72:	100.00 PERCENT.
EQUIPMENT #	73:	100.00 PERCENT.
EQUIPMENT #	74:	100.00 PERCENT.
EQUIPMENT #	75:	100.00 PERCENT.
EQUIPMENT #	76:	100.00 PERCENT.
EQUIPMENT #	77:	100.00 PERCENT.
EQUIPMENT #	78:	100.00 PERCENT.
EQUIPMENT #	79:	100.00 PERCENT.
EQUIPMENT #	80:	100.00 PERCENT.
EQUIPMENT #	81:	100.00 PERCENT.
EQUIPMENT #	82:	100.00 PERCENT.
EQUIPMENT #	83:	.00 PERCENT.
EQUIPMENT #	84:	.00 PERCENT.
EQUIPMENT #	85:	.00 PERCENT.
EQUIPMENT #	86:	.00 PERCENT.
EQUIPMENT #	87:	.00 PERCENT.
EQUIPMENT #	88:	.00 PERCENT.
EQUIPMENT #	89:	.00 PERCENT.
EQUIPMENT #	90:	.00 PERCENT.
EQUIPMENT #	91:	100.00 PERCENT.
EQUIPMENT #	92:	100.00 PERCENT.
EQUIPMENT #	93:	100.00 PERCENT.
EQUIPMENT #	94:	100.00 PERCENT.
EQUIPMENT #	95:	.00 PERCENT.
EQUIPMENT #	96:	.00 PERCENT.
GROUP #	100:	100.00 PERCENT.
GROUP #	101:	40.00 PERCENT.
GROUP #	102:	100.00 PERCENT.
GROUP #	103:	100.00 PERCENT.
GROUP #	104:	100.00 PERCENT.
GROUP #	105:	40.00 PERCENT.
GROUP #	106:	40.00 PERCENT.
GROUP #	107:	100.00 PERCENT.
GROUP #	108:	100.00 PERCENT.
GROUP #	109:	100.00 PERCENT.
GROUP #	110:	100.00 PERCENT.
GROUP #	111:	40.00 PERCENT.
GROUP #	112:	40.00 PERCENT.
GROUP #	113:	100.00 PERCENT.
GROUP #	114:	100.00 PERCENT.
GROUP #	115:	100.00 PERCENT.
GROUP #	116:	100.00 PERCENT.
GROUP #	117:	100.00 PERCENT.
GROUP #	118:	100.00 PERCENT.
GROUP #	119:	100.00 PERCENT.
GROUP #	120:	100.00 PERCENT.
GROUP #	121:	.00 PERCENT.
GROUP #	122:	100.00 PERCENT.
GROUP #	123:	100.00 PERCENT.
GROUP #	124:	100.00 PERCENT.
GROUP #	125:	100.00 PERCENT.
GROUP #	126:	100.00 PERCENT.
GROUP #	127:	100.00 PERCENT.
GROUP #	128:	.00 PERCENT.

GROUP #	129:	100.00 PERCENT.
GROUP #	130:	.00 PERCENT.
GROUP #	131:	20.00 PERCENT.
GROUP #	132:	.00 PERCENT.
GROUP #	133:	.00 PERCENT.
GROUP #	134:	100.00 PERCENT.
GROUP #	135:	.00 PERCENT.
GROUP #	136:	.00 PERCENT.
GROUP #	137:	.00 PERCENT.
GROUP #	138:	100.00 PERCENT.
GROUP #	139:	100.00 PERCENT.
GROUP #	140:	100.00 PERCENT.
GROUP #	141:	100.00 PERCENT.
GROUP #	142:	100.00 PERCENT.
GROUP #	143:	100.00 PERCENT.
GROUP #	144:	100.00 PERCENT.
GROUP #	145:	100.00 PERCENT.
GROUP #	146:	100.00 PERCENT.
GROUP #	147:	100.00 PERCENT.
GROUP #	148:	100.00 PERCENT.
GROUP #	149:	100.00 PERCENT.
GROUP #	150:	100.00 PERCENT.
GROUP #	151:	100.00 PERCENT.
GROUP #	152:	100.00 PERCENT.
GROUP #	153:	20.00 PERCENT.
GROUP #	154:	100.00 PERCENT.
GROUP #	155:	.00 PERCENT.

THE COST OF THE SYSTEM IS $ 147060.00

EQUIPMENT #	1	SURV COST INDEX:	6.8
EQUIPMENT #	2	SURV COST INDEX:	6.8
EQUIPMENT #	3	SURV COST INDEX:	6.8
EQUIPMENT #	4	SURV COST INDEX:	6.8
EQUIPMENT #	5	SURV COST INDEX:	6.8
EQUIPMENT #	6	SURV COST INDEX:	6.8
EQUIPMENT #	7	SURV COST INDEX:	2.7
EQUIPMENT #	8	SURV COST INDEX:	2.7
EQUIPMENT #	9	SURV COST INDEX:	2.7
EQUIPMENT #	10	SURV COST INDEX:	6.8
EQUIPMENT #	11	SURV COST INDEX:	6.8
EQUIPMENT #	12	SURV COST INDEX:	6.8
EQUIPMENT #	13	SURV COST INDEX:	6.8
EQUIPMENT #	14	SURV COST INDEX:	6.8
EQUIPMENT #	15	SURV COST INDEX:	6.8
EQUIPMENT #	16	SURV COST INDEX:	1.4
EQUIPMENT #	17	SURV COST INDEX:	.0
EQUIPMENT #	18	SURV COST INDEX:	.0
EQUIPMENT #	19	SURV COST INDEX:	6.8
EQUIPMENT #	20	SURV COST INDEX:	6.8
EQUIPMENT #	21	SURV COST INDEX:	6.8
EQUIPMENT #	22	SURV COST INDEX:	6.8
EQUIPMENT #	23	SURV COST INDEX:	6.8
EQUIPMENT #	24	SURV COST INDEX:	6.8
EQUIPMENT #	25	SURV COST INDEX:	6.8
EQUIPMENT #	26	SURV COST INDEX:	6.8
EQUIPMENT #	27	SURV COST INDEX:	6.8
EQUIPMENT #	28	SURV COST INDEX:	6.8
EQUIPMENT #	29	SURV COST INDEX:	6.8
EQUIPMENT #	30	SURV COST INDEX:	6.8
EQUIPMENT #	31	SURV COST INDEX:	.0
EQUIPMENT #	32	SURV COST INDEX:	6.8
EQUIPMENT #	33	SURV COST INDEX:	6.8
EQUIPMENT #	34	SURV COST INDEX:	.0

EQUIPMENT #	35	SURV COST INDEX:	6.8
EQUIPMENT #	36	SURV COST INDEX:	6.8
EQUIPMENT #	37	SURV COST INDEX:	6.8
EQUIPMENT #	38	SURV COST INDEX:	6.8
EQUIPMENT #	39	SURV COST INDEX:	6.8
EQUIPMENT #	40	SURV COST INDEX:	6.8
EQUIPMENT #	41	SURV COST INDEX:	6.8
EQUIPMENT #	42	SURV COST INDEX:	6.8
EQUIPMENT #	43	SURV COST INDEX:	6.8
EQUIPMENT #	44	SURV COST INDEX:	6.8
EQUIPMENT #	45	SURV COST INDEX:	6.8
EQUIPMENT #	46	SURV COST INDEX:	6.8
EQUIPMENT #	47	SURV COST INDEX:	6.8
EQUIPMENT #	48	SURV COST INDEX:	6.8
EQUIPMENT #	49	SURV COST INDEX:	6.8
EQUIPMENT #	50	SURV COST INDEX:	6.8
EQUIPMENT #	51	SURV COST INDEX:	6.8
EQUIPMENT #	52	SURV COST INDEX:	6.8
EQUIPMENT #	53	SURV COST INDEX:	6.8
EQUIPMENT #	54	SURV COST INDEX:	6.8
EQUIPMENT #	55	SURV COST INDEX:	6.8
EQUIPMENT #	56	SURV COST INDEX:	6.8
EQUIPMENT #	57	SURV COST INDEX:	6.8
EQUIPMENT #	58	SURV COST INDEX:	6.8
EQUIPMENT #	59	SURV COST INDEX:	6.8
EQUIPMENT #	60	SURV COST INDEX:	6.8
EQUIPMENT #	61	SURV COST INDEX:	6.8
EQUIPMENT #	62	SURV COST INDEX:	6.8
EQUIPMENT #	63	SURV COST INDEX:	6.8
EQUIPMENT #	64	SURV COST INDEX:	6.8
EQUIPMENT #	65	SURV COST INDEX:	6.8
EQUIPMENT #	66	SURV COST INDEX:	6.8
EQUIPMENT #	67	SURV COST INDEX:	6.8
EQUIPMENT #	68	SURV COST INDEX:	.0
EQUIPMENT #	69	SURV COST INDEX:	.0
EQUIPMENT #	70	SURV COST INDEX:	6.8
EQUIPMENT #	71	SURV COST INDEX:	6.8
EQUIPMENT #	72	SURV COST INDEX:	6.8
EQUIPMENT #	73	SURV COST INDEX:	6.8
EQUIPMENT #	74	SURV COST INDEX:	6.8
EQUIPMENT #	75	SURV COST INDEX:	6.8
EQUIPMENT #	76	SURV COST INDEX:	6.8
EQUIPMENT #	77	SURV COST INDEX:	6.8
EQUIPMENT #	78	SURV COST INDEX:	6.8
EQUIPMENT #	79	SURV COST INDEX:	6.8
EQUIPMENT #	80	SURV COST INDEX:	6.8
EQUIPMENT #	81	SURV COST INDEX:	6.8
EQUIPMENT #	82	SURV COST INDEX:	6.8
EQUIPMENT #	83	SURV COST INDEX:	.0
EQUIPMENT #	84	SURV COST INDEX:	.0
EQUIPMENT #	85	SURV COST INDEX:	.0
EQUIPMENT #	86	SURV COST INDEX:	.0
EQUIPMENT #	87	SURV COST INDEX:	.0
EQUIPMENT #	88	SURV COST INDEX:	.0
EQUIPMENT #	89	SURV COST INDEX:	.0
EQUIPMENT #	90	SURV COST INDEX:	.0
EQUIPMENT #	91	SURV COST INDEX:	6.8
EQUIPMENT #	92	SURV COST INDEX:	6.8
EQUIPMENT #	93	SURV COST INDEX:	6.8
EQUIPMENT #	94	SURV COST INDEX:	6.8
EQUIPMENT #	95	SURV COST INDEX:	.0
EQUIPMENT #	96	SURV COST INDEX:	.0
GROUP #	100	SURV COST INDEX:	6.8
GROUP #	101	SURV COST INDEX:	2.7

GROUP #	102	SURV COST INDEX:	6.8
GROUP #	103	SURV COST INDEX:	6.8
GROUP #	104	SURV COST INDEX:	6.8
GROUP #	105	SURV COST INDEX:	2.7
GROUP #	106	SURV COST INDEX:	2.7
GROUP #	107	SURV COST INDEX:	6.8
GROUP #	108	SURV COST INDEX:	6.8
GROUP #	109	SURV COST INDEX:	6.8
GROUP #	110	SURV COST INDEX:	6.8
GROUP #	111	SURV COST INDEX:	2.7
GROUP #	112	SURV COST INDEX:	2.7
GROUP #	113	SURV COST INDEX:	6.8
GROUP #	114	SURV COST INDEX:	6.8
GROUP #	115	SURV COST INDEX:	6.8
GROUP #	116	SURV COST INDEX:	6.8
GROUP #	117	SURV COST INDEX:	6.8
GROUP #	118	SURV COST INDEX:	6.8
GROUP #	119	SURV COST INDEX:	6.8
GROUP #	120	SURV COST INDEX:	6.8
GROUP #	121	SURV COST INDEX:	.0
GROUP #	122	SURV COST INDEX:	6.8
GROUP #	123	SURV COST INDEX:	6.8
GROUP #	124	SURV COST INDEX:	6.8
GROUP #	125	SURV COST INDEX:	6.8
GROUP #	126	SURV COST INDEX:	6.8
GROUP #	127	SURV COST INDEX:	6.8
GROUP #	128	SURV COST INDEX:	.0
GROUP #	129	SURV COST INDEX:	6.8
GROUP #	130	SURV COST INDEX:	.0
GROUP #	131	SURV COST INDEX:	1.4
GROUP #	132	SURV COST INDEX:	.0
GROUP #	133	SURV COST INDEX:	.0
GROUP #	134	SURV COST INDEX:	6.8
GROUP #	135	SURV COST INDEX:	.0
GROUP #	136	SURV COST INDEX:	.0
GROUP #	137	SURV COST INDEX:	.0
GROUP #	138	SURV COST INDEX:	6.8
GROUP #	139	SURV COST INDEX:	6.8
GROUP #	140	SURV COST INDEX:	6.8
GROUP #	141	SURV COST INDEX:	6.8
GROUP #	142	SURV COST INDEX:	6.8
GROUP #	143	SURV COST INDEX:	6.8
GROUP #	144	SURV COST INDEX:	6.8
GROUP #	145	SURV COST INDEX:	6.8
GROUP #	146	SURV COST INDEX:	6.8
GROUP #	147	SURV COST INDEX:	6.8
GROUP #	148	SURV COST INDEX:	6.8
GROUP #	149	SURV COST INDEX:	6.8
GROUP #	150	SURV COST INDEX:	6.8
GROUP #	151	SURV COST INDEX:	6.8
GROUP #	152	SURV COST INDEX:	6.8
GROUP #	153	SURV COST INDEX:	1.4
GROUP #	154	SURV COST INDEX:	6.8
GROUP #	155	SURV COST INDEX:	.0

Appendix C. BEAVER Source Code

```
C===================================================================
C==                                                  == 5/7/92 ==
C==      - GATOR 2.0 -                               ============
C==      INITIAL RELEASE: JUNE 14,1984                           ==
C==                                                              ==
C==      - BEAVER -                                              ==
C==      MAJOR REVISION:  MAY     1992                           ==
C==                                                              ==
C==   THE ENHANCEMENTS HERE ARE FOR THE SURVIVABILITY FUNCTION   ==
C==   ADDED TO THE GATOR PROGRAM.  IT IS INTENDED TO BE AN ADDED ==
C==   FEATURE WHICH IS SELECTED AS AN ANALYSIS IN THE INPUT      ==
C==   FILE.  THE ANALYSIS ADDS SOME LINES TO THE EXISTING        ==
C==   PROGRAM, BUT MOST OF THE ADDITION IS IN THE                ==
C==   SUBROUTINES ADDED AS OF THE MOD DATE.                      ==
C==                                                              ==
C==   NOTE:  THIS VERSION IS SET UP TO NEED DIMENSIONING         ==
C==          OF ARRAY LENGTHS IN ARRAYDIM.INC                    ==
C==                                                              ==
C==          UP(I) = 0  ==> NO FAILURE                           ==
C==                = 1  ==> FAILURE                              ==
C==                                                              ==
C===================================================================
C===================================================================
C
C-----    FILE ARRAYDIM.INC CONTAINS THE ARRAY DIMENSIONS AND
C         FILE DECLARE.INC CONTAINS GLOBAL DECLARATIONS
C
          INCLUDE ARRAYDIM.INC
          INCLUDE DECLARE.INC
C
C-------------------------------------------------------------------
C
C-----    SET I/O UNITS
C
          INPUT  =  7
          OUTPUT = 11
          CONSOL =  9
          AUX1   =  8
          AUX2   =  3
          AUX3   = 12
C
          CALL OPENF
C
C-----    ZERO OUT ARRAYS NOT INITIALIZED IN INITAL
C
          CALL ZZERO
C
C-----    PRINT HEADER
C
          CALL HEADER
C
C-----    READ IN THE INPUT DATA
C
          CALL READIT
C
```

```
C-----    INITIALIZE STATISTICS ACROSS THE MISSIONS
C
          NEVENT = 0
          NFT    = 0
          NREP   = 0
          NTFALS = 0
          ISAVE  = 0
C
          PFT    = 0.0
          RSQ    = 0.0
          RSUM   = 0.0
          SFT    = 0.0
          SFT2   = 0.0
C
C-----    SET NUMBER OF SCREEN LINES FOR "PAUSING" THE RUN
C
          LINES  = 20
          NEWSCR = LINES * MISSPR
C
       CALL POWERPATH
C
C -----      CALCULATE THE RADIUS FROM THE HIT LOCATION TO EACH COMPONENT.
C -----      SKIP THE RADIUS CALCS IF THE SURVIVABILITY ANALYSIS ISN'T
C -----      GOING TO BE RUN.
C
       IF ( SURVIVE .EQ. 0 ) GOTO 88
C
C -----      GET THE HIT COORDINATES AND GET THEM READY FOR USE.
C
C
       HITX = HITDATA(1)
       HITY = HITDATA(2)
       HITZ = HITDATA(3)
       WRITE(OUTPUT, 9699)HITX, HITY, HITZ
       WRITE(CONSOL, 9699)HITX, HITY, HITZ
 9699  FORMAT(/'HIT LOCATION IS ',3F8.2/)
C
C -----      FIND THE RADIUS FROM THE HIT TO THE COMPONENT FOR EACH OF
C -----      THE PIECES OF THE SYSTEM.
C
       DO 40 I = 1, NPARTS
C
C -----      IF THE EQuipment TYPE READ IN IS A CABLE ( ie 5 ) THEN THE
C -----      DISTANCE TO THE CLOSEST POINT MUST BE CALCULATED CONSIDERING
C -----      BOTH END POINTS AND THE ORIENTATION OF THE LINE SEGMENT TO THE
C -----      HIT POINT.
C
             IF   ( EQTYPE( I ) .EQ. 5 ) THEN
                   CX1 = EQDATA (I,1)
                   CY1 = EQDATA (I,2)
                   CZ1 = EQDATA (I,3)
                   CX2 = EQDATA (I,4)
                   CY2 = EQDATA (I,5)
                   CZ2 = EQDATA (I,6)
C
                   CALL RCABLE (I)
C
C -----      RCABLE SETS THE RADIUS (I) ARRAY VALUE FOR THE CABLES.
```

```
C
                  GOTO 38
            ELSE
                  CXX = EQDATA (I,1)
                  CYY = EQDATA (I,2)
                  CZZ = EQDATA (I,3)
            ENDIF
C
            RADSQ = (HITX-CXX)**2+(HITY-CYY)**2+
     &                    (HITZ-CZZ)**2
            RAD = SQRT(RADSQ)
            RADIUS ( I ) = RAD
C
 38   WRITE(OUTPUT, 13600) I, RADIUS(I)
13600 FORMAT('I = ',I4,' RADIUS = ',F10.2)
 40   CONTINUE
         GOTO 89
C
C------------------------------------------------------------------------
C-----   S T A R T   O F   S I M U L A T I O N   -------------------
C------------------------------------------------------------------------
C
 88      WRITE (OUTPUT,90120)
 89      WRITE (CONSOL,90110)
C
 90      DO 10000 MNO = 1,NMISS
C
C-----      INITIALIZE FOR EACH MISSION
C
         CALL INITAL
         NTRIAL = 0
C
C
      CALL LINEUP
C
C------------------------------------------------------------------------
C-----   T O P   O F   E V E N T   L O O P   ----------------------
C------------------------------------------------------------------------
C
C-----      UPDATE PROBABILITY DISTRIBUTION FUNCTION;
C-----      GENERATE NEXT EVENT TIME, TYPE AND BLOCK
C
  100       CONTINUE
C
C-----      THIS LINE BRANCHES FOR SURVIVABILITY ANALYSIS
C
         IF (SURVIVE .EQ. 1) GOTO 7000
C
            CALL GENVNT(K,BLK)
C
            NEVENT = NEVENT + 1
            ISAVE  = UP(ISYS)
C
C-----   BRANCH ON EVENT CODE [1=DETECTED FAILURE 2=REPAIR
C        3=PHASE CHANGE]
C
            GO TO (1000, 2000, 3000), K
C
```

```
C-----------------------------------------------------
C  CASE  SYSTEM STATUS TRANSITION HANDLING:
C
C   1    UP           -> UP:    DO =>  NOTHING
C   2    UP           -> DOWN:  DO =>  BAKST1 -> BAKST2
C   3    DOWN         -> UP:    DO =>  BAKST3
C   4    DOWN         -> DOWN:  DO =>  BAKST3 -> BAKST1
C-----------------------------------------------------
C
C-----   FAILURE OF EQUIPMENT NUMBER BLK
C
 1000       CALL FAIL(BLK)
            CALL STAT(BLK,1)
            GOTO 100
C
C-----   REPAIR EQUIPMENT NUMBER BLK (BLK = SRVBLK IN THIS CASE)
C
 2000       CALL REPAIR (BLK)
            CALL STAT (BLK,0)
            GOTO 100
C
C-----      SECTION WHICH CONTROLS THE SURVIVABILITY ANALYSIS.
C-----      UPON RETURNING FROM THE DAMAGE ROUTINE, THE SINGLE PHASE
C-----      OF THE MISSION CAUSES PCHANG TO TERMINATE THE MISSION
C-----      AND CLOSE OUT THE STATISTICS.  THE PROGRAM WILL THEN
C-----      CONTINUE TO LOOP UNTIL THE SPECIFIED NUMBER OF MISSIONS
C-----      IS EVALUATED.
C
 7000    NEVENT=NEVENT+1
         ISAVE=UP(ISYS)
C
C
         CALL DAMAGE
C
C
      CALL POWERCHK
C
C-----   PHASE CHANGE EVENT
C
 3000       CALL PCHANG
            IF ( .NOT. DONE ) GOTO 100
            CALL SYS_STATS
C
C--------------------------------------------------------------------
C-----   E N D  O F  E V E N T  L O O P  ----------------------------
C--------------------------------------------------------------------
C
C-----      CLOSE OUT STATISTICS FOLLOWING THE END OF CURRENT MISSION
C
            DO 4000 I = 1,NEQT
               IF ( SPARES(I) .EQ. 0 ) NSNOT(I) = NSNOT(I) + 1
               N3(I) = N3(I) + SPARE1(I) - SPARES(I)
 4000       CONTINUE
C
            DO 4100 I = 1,NPARTS
               IF ( NUMFAL (I) .NE. 0 ) N2(I) = N2(I) + 1
               IF ( UP(I) .NE. 0 ) CDWNTM(I) = CDWNTM(I)
     &                                  + TIME - DWNTME(I)
```

```
            N1(I) = N1(I) + NUMFAL(I)
            DUM = DURMIS - CDWNTM(I)
            T1(I) = T1(I) + DUM
            SST1(I) = SST1(I) + DUM * DUM
 4100     CONTINUE
C
          DO 4200 I = LOWGRP,NBLKS
            IF ( NUMFAL(I) .NE. 0 ) N2(I) = N2(I) + 1
            IF ( UP(I). NE. 0 ) CDWNTM(I) = CDWNTM(I)
     &                                   + TIME - DWNTME(I)
            N1(I) = N1(I) + NUMFAL(I)
            DUM = DURMIS - CDWNTM(I)
            T1(I) = T1(I) + DUM
            SST1(I) = SST1(I) + DUM * DUM
 4200     CONTINUE
C
C -----   SWITCH TO SKIP SOME ANALYSIS NOT NEEDED
C
 3600 IF ( SURVIVE .EQ. 1 ) GOTO 9000
C
C-----    COMPUTE SYSTEM RELIABILITY AND AVAILABILITY
C         AND PRINT OUT EVERY 'MISSPR' MISSIONS
C
          IF ( MNO .EQ. NMISS )          GOTO 5000
          IF ( MOD(MNO,MISSPR) .NE. 0 ) GOTO 10000
C
 5000     RMNO = 1.0 / FLOAT(MNO)
          R = 1.0 - FLOAT(NFT)   * RMNO
          A = T1(ISYS) / DURMIS * RMNO
          S = SQRT( R * (1 - R) * RMNO )
C
          SNSDEV = NSDEV * S
          LLIM = R - SNSDEV
          ULIM = R + SNSDEV
          WRITE (CONSOL,90010) MNO,R,S,A
          WRITE (OUTPUT,90020) MNO,R,S,LLIM,ULIM,A
C
          IF ( MNO .EQ. NMISS )          GOTO 10000
          IF ( MOD(MNO,NEWSCR) .NE. 0 ) GOTO 10000
          WRITE (CONSOL,90025)
          PAUSE
C
 9000 IF ( SURVIVE .EQ. 0 ) GOTO 10000
C
C -----   MISSION EVALUATION.  SNUMFAL IS THE SURVIVABILITY
C -----   ANALYSIS NUMBER OF FAILURES
C
C
      DO 9200 I = 1, NBLKS
        TSNUMFAL = SNUMFAL(I)
        SNUMFAL(I) = TSNUMFAL + FLOAT(NUMFAL(I))
 9200 CONTINUE
C
10000    CONTINUE
C
C---------------------------------------------------------------------
C-----   E N D  O F  S I M U L A T I O N   ----------------------------
C---------------------------------------------------------------------
```

```
C
          WRITE (CONSOL,90030) NTFALS,NEVENT
          WRITE (CONSOL,90025)
          PAUSE
C
          CONTINUE
C
C -----      SWITCH OVER THE NORMAL REPORT FOR SURVIVABILITY ANALYSIS
C
11010 IF ( SURVIVE .EQ. 0 ) GOTO 11020
C
C -----      REPSURV IS THE REPort for SURVivability analysis SUBROUTINE
C
11012 CALL REPSURV
11014 GOTO 12500
C
C-----   GENERATE FINAL SIMULATION REPORT
C
C-----   PRINT OUT SYSTEM FIRST FAILURE INFORMATION
C
11020     IF ( KOPT.LT.5 )  GOTO 12000
          IF ( NFT.GT.1 )   GOTO 11200
          IF ( NFT.EQ.1 )   GOTO 11100
          WRITE (  AUX3,90040)
          GOTO 12000
C
11100     WRITE (  AUX3,90050) TFFT(1)
          GOTO 12000
C
11200     DPR   = 1.0 / FLOAT(NFT)
          ATBF  = SFT * DPR
          STDBF = SQRT( ( SFT2 - SFT * SFT * DPR ) / (NFT - 1) )
          WRITE (  AUX3,90060) ATBF,STDBF,NFT
C
          CALL HSORT(TFFT,NFT)
C
          WRITE (  AUX3,90070)
          CPROB = 0.0
C
          DO 11300 KK = 1,NFT
             CPROB = CPROB + DPR
             WRITE (  AUX3,90080) KK,CPROB,TFFT(KK)
11300     CONTINUE
C
C-----   GENERATE FINAL SIMULATION REPORT
C
C
12000     CALL REPORT
C
C
12500 CONTINUE
C
C-----   CLOSE I/O UNITS
C
          CALL CLOSEF
C
          STOP
C
```

```
90010     FORMAT(' MISSION : ',I6,'   RELIABILITY: ',F7.4,' +/- ',E10.4,
     &          '    AVAILABILITY: ',F7.4)
90020     FORMAT(T6,I6,T15,F7.4,T25,E10.4,T37,F7.4,T47,F7.4,
     &          T57,F7.4)
90025     FORMAT(' < HIT RETURN TO CONTINUE >')
90030     FORMAT(/,' ** SIMULATION COMPLETED ! ** ',
     &          /' NO. OF FAULTS:',I10,' NO. OF EVENTS PROCESSED:',I10)
90040     FORMAT(' NO SYSTEM FAILURES OCCURRED')
90050     FORMAT(' ONE FAILURE OCURRED AT : ',F20.10,' HOURS')
90060     FORMAT(/,T10,'MEAN TIME BETWEEN FAILS : ',T35,F20.5,
     1          /,T10,'STANDARD DEVIATION : ',T35,F20.5,
     2          /,T10,'NO. OF FAILS : ',T40,I10,/)
90070     FORMAT(' NO.     CUM. PROB.        TIME OF FIRST FAILURE',/)
90080     FORMAT(' ',I6,' ',F10.5,8X,F12.2)
90110     FORMAT(' ** SIMULATION STARTED...** ')
90120     FORMAT(' PAGE BREAK',/,T5,'MISSIONS',T16,'RELIAB',T27,'STD DEV',
     &          T39,'LO LIM',T49,'UP LIM',T59,'AVAIL')
C
          END
C
```

```
C ***********************************************************************
C ***********************************************************************
C ***********************************************************************
C
      SUBROUTINE BAKST1
C                                                              02/29/88
C-----------------------------------------------------------------------
C    PURPOSE:   TRAVERSES DOWNWARD FROM SYSTEM NODE THRU FAILED PATHS;
C               COUNTS NO. OF CULPABLE EQUIPMENTS AT BOTTOM OF NETWORK;
C               SETS FLAGS (SYSDWN) FOR SUBSEQUENT ROUTINE (BAKST2) TO
C               "NAVIGATE" BY. IN BAKST STACK POINTER MOVES DOWN
C               A VERTICAL BRANCH (FROM A GROUP TO ITS MEMBERS) TO ITS
C               END AND THEN MOVES HORIZONTALLY (WITHIN A GROUP).
C    CALLED IN: MAIN
C-----------------------------------------------------------------------
C
C-----    FILE ARRAYDIM.INC CONTAINS THE ARRAY DIMENSIONS AND
C         FILE DECLARE.INC CONTAINS GLOBAL DECLARATIONS
C
      INCLUDE ARRAYDIM.INC
      INCLUDE DECLARE.INC
C
C-----    LOCAL DECLARATIONS
C
      INTEGER     SP
C
C-----    END OF LOCAL DECLARATIONS
C
C-----------------------------------------------------------------------
C
C-----    INITIALIZE THE STACK POINTER
C
      SP     = 0
      NBAD   = 0
      BLK    = ISYS
      OPNTME = TIME
C
C-----    NOTE THAT SYSTEM DWNTME IS MARKED HERE
C
 1000 CONTINUE
C
C-----    SET FLAG FOR THIS BLOCK
C
      SYSDWN(BLK) = 1
      IBEG = RSTAR(BLK)
      IEND = RSTAR(BLK+1) - 1
      IND  = IBEG
C
C-----    NOTE THAT IEND+1 - IBEG = # ARCS LEAVING BLK GOING DOWNWARD
C
C-----    BLOCK IS AN EQUIPMENT
C
      IF ( IBEG .LE. IEND ) GOTO 1300
      NBAD = NBAD + 1
C
C-----    THE NUMBER OF CULPABLE EQUIPMENTS IS COUNTED HERE
C
      GOTO 1600
```

```
C
C-----  BLOCK IS A GROUP
C
C-----  CHECK FOR COMPLETION OF HORIZONTAL MOVE
C
 1300   IF ( IND .GT. IEND ) GOTO 1600
C
C-----  RECALL THAT RARC MOVES FROM TOP DOWN!
C
        TBLK = RARC(IND)
        IF ( UP(TBLK) .EQ. 0 )     GOTO 1900
        IF ( SYSDWN(TBLK) .EQ. 1 ) GOTO 1900
        SP = SP + 4
        IF ( SP .GT. LEN12 ) THEN
        WRITE(OUTPUT,1400)
        WRITE(CONSOL,1400)
 1400      FORMAT(/' ERROR FOUND IN  SUBROUTINE BAKST1')
        CALL ERRKDE(9,SP,LEN12)
      ENDIF
        STAK(SP-3) = BLK
        STAK(SP-2) = IBEG
        STAK(SP-1) = IEND
        STAK(SP)   = IND
        BLK = TBLK
        GOTO 1000
C
C-----  UPWARD RETRACE
C
 1600   IF ( SP .EQ. 0 ) RETURN
        TBLK = BLK
        IND  = STAK(SP)
        IEND = STAK(SP-1)
        IBEG = STAK(SP-2)
        BLK  = STAK(SP-3)
        SP = SP - 4
C
C-----  INCREMENT INDEX OF ARC'S TAIL TO CONTINUE HORIZONTAL MOVE
C
 1900   IND = IND + 1
        GOTO 1300
C
        END
C
```

```
C *********************************************************************
C *********************************************************************
C *********************************************************************
C
      SUBROUTINE BAKST2
C                                                            02/29/88
C---------------------------------------------------------------------
C   PURPOSE:   TRAVERSES FROM SYSTEM NODE ALONG FLAGS (SYSDWN) SET IN
C              BAKST1. (SEE COMMENTS ON STACK POINTER IN BAKST1);

C              ADDS 1/NBAD TO HIT LIST OF CULPABLE EQUIPMENTS.
C   CALLED IN: MAIN
C---------------------------------------------------------------------
C
C-----  FILE ARRAYDIM.INC CONTAINS THE ARRAY DIMENSIONS AND
C       FILE DECLARE.INC CONTAINS GLOBAL DECLARATIONS
C
      INCLUDE ARRAYDIM.INC
      INCLUDE DECLARE.INC
C
C-----  LOCAL DECLARATIONS
C
      INTEGER    SP
C
C-----  END OF LOCAL DECLARATIONS
C
C---------------------------------------------------------------------
C
C-----  IADFLG IS USED TO MARK EQUIPMENTS ALREADY UPDATED
C       IN THE RELIABILITY HIT LIST
C
      DO 10 I = 1,NPARTS
         IADFLG(I) = 0
   10 CONTINUE
C
C-----  INITIALIZE THE STACK POINTER
C
      SP = 0
      BLK = ISYS
      IF (NBAD .GT. O) RNBAD = 1.0 / NBAD
C
 1000 CONTINUE
      IBEG = RSTAR(BLK)
      IEND = RSTAR(BLK+1) - 1
      IND  = IBEG
C
C-----  BLOCK IS AN EQUIPMENT
C
      IF ( IBEG .LE. IEND ) GOTO 1300
C
C-----  ADD 1/NBAD TO RELIABILITY HIT LIST
C
      IF ( IADFLG(BLK) .NE. 0 ) GOTO 1600
      IADFLG(BLK) = 1
      SYSCRT(BLK) = SYSCRT(BLK) + RNBAD
      GOTO 1600
C
C-----  BLOCK IS A GROUP
```

```
C
 1300     IF ( IND .GT. IEND ) GOTO 1600
          TBLK = RARC(IND)
          IF ( SYSDWN(TBLK) .EQ. 0 ) GOTO 1900
          SP = SP + 4
          IF ( SP .GT. LEN12) THEN
          WRITE(OUTPUT,1400)
          WRITE(CONSOL,1400)
 1400        FORMAT(/' ERROR FOUND IN  SUBROUTINE BAKST2')
          CALL ERRKDE(10,SP,LEN12)
          ENDIF
          STAK(SP-3) = BLK
          STAK(SP-2) = IBEG
          STAK(SP-1) = IEND
          STAK(SP)   = IND
          BLK = TBLK
          GOTO 1000
C
C-----    UPWARD RETRACE
C
 1600     IF ( SP .EQ. 0 ) RETURN
          TBLK = BLK
          IND  = STAK(SP)
          IEND = STAK(SP-1)
          IBEG = STAK(SP-2)
          BLK  = STAK(SP-3)
          SP   = SP - 4
C
C-----    INCREMENT INDEX OF ARC'S TAIL
C
 1900     IND = IND + 1
          GOTO 1300
C
          END
C
```

```
C ***********************************************************************
C ***********************************************************************
C ***********************************************************************
C
      SUBROUTINE BAKST3
C                                                              2/29/88
C-----------------------------------------------------------------------
C    PURPOSE:   TRAVERSES FROM SYSTEM NODE ALONG FLAGS (SYSDWN) SET IN
C               BAKST1. CLOSES OUT EXISTING TIME AVERAGES FOR
C               AVAILABILITY HIT LIST WITH EXISTING NBAD AND RESETS
C               FLAGS (SYSDWN) ALONG THE WAY.
C               (SEE COMMENTS ON STACK POINTER IN BAKST1)
C    CALLED IN: MAIN
C-----------------------------------------------------------------------
C
C-----    FILE ARRAYDIM.INC CONTAINS THE ARRAY DIMENSIONS AND
C         FILE DECLARE.INC CONTAINS GLOBAL DECLARATIONS
C
      INCLUDE ARRAYDIM.INC
      INCLUDE DECLARE.INC
C
C-----    LOCAL DECLARATIONS
C
      INTEGER     SP
C
C-----    END OF LOCAL DECLARATIONS
C
C-----------------------------------------------------------------------
C
C-----    INITIALIZE THE STACK POINTER
C
      SP = 0
      BLK = ISYS
C
C-----    THE TIME DURING WHICH THE SYSTEM IS UNAVAILABLE
C         IS DIVIDED AMONG ALL CULPRIT EQUIPMENTS
C
      IF (NBAD .GT. O) DELAV = ( TIME - OPNTME ) / NBAD
C
 1000 CONTINUE
C
C-----    RESET FLAG FOR THIS BLOCK
C
      SYSDWN(BLK) = 0
      IBEG = RSTAR(BLK)
      IEND = RSTAR(BLK+1) - 1
      IND  = IBEG
C
C-----    BLOCK IS AN EQUIPMENT
C
      IF ( IBEG .LE. IEND ) GOTO 1300
      AVCUM(BLK)  = AVCUM(BLK)  + DELAV
      IF ( SPARES(TYPE(BLK)) .LE. 0 )
     & AVSPAR(BLK) = AVSPAR(BLK) + DELAV
      GOTO 1600
C
C-----    BLOCK IS A GROUP
C
```

```
1300    IF ( IND .GT. IEND ) GOTO 1600
        TBLK = RARC(IND)
        IF ( SYSDWN(TBLK) .EQ. 0) GOTO 1900
        SP = SP + 4
        IF ( SP .GT. LEN12) THEN
        WRITE(OUTPUT,1400)
        WRITE(CONSOL,1400)
1400       FORMAT(/' ERROR FOUND IN  SUBROUTINE BAKST3')
        CALL ERRKDE(11,SP,LEN12)
        ENDIF
        STAK(SP-3) = BLK
        STAK(SP-2) = IBEG
        STAK(SP-1) = IEND
        STAK(SP)   = IND
        BLK = TBLK
        GOTO 1000
C
C-----  UPWARD RETRACE
C
 1600   IF ( SP .EQ. 0 ) RETURN
C
        TBLK = BLK
        IND  = STAK(SP)
        IEND = STAK(SP-1)
        IBEG = STAK(SP-2)
        BLK  = STAK(SP-3)
        SP   = SP - 4
C
C-----  INCREMENT INDEX OF ARC'S TAIL
C
 1900   IND = IND + 1
        GOTO 1300
C
        END
C
```

```
C *********************************************************************
C *********************************************************************
C *********************************************************************
C
        SUBROUTINE CHKLOAD
C                                                               5/6/92
C---------------------------------------------------------------------
C    PURPOSE:   TRAVERSES DOWNWARD FROM LOAD NODES THRU ALL PATHS
C               TO DETERMINE WHICH LOADS ARE POWERED, WHERE THE POWER
C               COMES FROM, AND WHAT THE LOADING IS ON THE GENERATORS
C
C
C
C    CALLED IN: POWERCHK
C---------------------------------------------------------------------
C
C-----   FILE ARRAYDIM.INC CONTAINS THE MAXIMUM ARRAY DIMENSIONS AND
C        FILE DECLARE.INC  CONTAINS GLOBAL DECLARATIONS
C
        INCLUDE ARRAYDIM.INC
        INCLUDE DECLARE.INC
C
C-----   LOCAL DECLARATIONS
C
        INTEGER     SP
C
C-----   END OF LOCAL DECLARATIONS
C
C---------------------------------------------------------------------
C
C
C -----     ZERO THE ARRAYS
C
  100 DO 1000 I = 1, NUMSRC
        DO 1010 J = 1, NUMLD
          GENLOAD(I,J) = 0.0
 1010   CONTINUE
 1000 CONTINUE
C
      DO 1020 I = 1, LEN17
        SRCLD(I) = 0.0
        OVRLD(I) = 0.0
 1020 CONTINUE
C
C -----     LOADWN USED TO INDICATE LOADS WHICH ARE OK, BUT ARE NOT POWERED
C -----     AS INDICATED BY THE LACK OF FINDING A CONTINUOUS POWER PATH
C -----     FROM THE SOURCE TO THE LOAD.
C
      DO 1030 I = 1, LEN18
        PLLCHK(I) = 0.0
        LOADWN(I) = 0
 1030 CONTINUE
C
        DO 9973 J=1,NUMLD
C
C-----  INITIALIZE THE STACK POINTER
C
        SP = 0
```

```
         BLK = LOAD(J)
C
      WRITE(CONSOL, 90910) BLK
      WRITE(OUTPUT, 90910) BLK
90910 FORMAT('LOOKING FOR ALL SOURCES FOR LOAD ',I4)
      IND = FSTAR(BLK)
      BLK = ARC(IND)
C
C -----     IF THE LOAD'S PARENT SUBGROUP IS DOWN, DON'T BOTHER TO CHECK
C
      IF (UP(BLK) .EQ. 1) GOTO 9973
C
C
 9910    CONTINUE
         IBEG = RSTAR(BLK)
         IEND = RSTAR(BLK+1) - 1
         IND  = IBEG
C
C-----   BLOCK IS AN EQUIPMENT, SO GO TO 9960
C
         IF ( IBEG .LE. IEND ) GOTO 9930
C
         GOTO 9960
C
C-----   BLOCK IS A GROUP
C
C -----     CHECK FOR COMPLETION OF HORIZONTAL MOVE
C
 9930    IF ( IND .GT. IEND ) GOTO 9960
C
         TBLK = RARC(IND)
C
C
C -----     IF BLK IS DOWN, NO PATH FROM LOAD TO SOURCE HERE, SO TRY .
C -----     ANOTHER PATH...
C
      IF ( UP(TBLK) .EQ. 1 ) THEN
        GOTO 9990
      ELSE
      ENDIF
C
         SP = SP + 4
         IF ( SP .GT. LEN12 ) THEN
           WRITE (OUTPUT,9940)
           WRITE (CONSOL,9940)
 9940      FORMAT(/' ERROR FOUND IN  SUBROUTINE CHKLOAD')
           CALL ERRKDE(10,SP,LEN12)
         ENDIF
         STAK(SP-3) = BLK
         STAK(SP-2) = IBEG
         STAK(SP-1) = IEND
         STAK(SP)   = IND
         BLK = TBLK
         GOTO  9910
C
C-----   UPWARD RETRACE
C
 9960    IF ( SP .EQ. 0 ) GOTO 9999
```

```
C
C
C -----     IF THE BLK IS DOWN, GO ON TO CHECK OTHER EQUIPMENT
C
      IF ( UP(BLK) .EQ. 1 ) THEN
        GOTO 9972
      ELSE
      ENDIF
C
      DO 9971 I=1,NUMSRC
      IF (BLK .EQ. SOURCE(I)) THEN
         WRITE(CONSOL,90900) BLK
         WRITE(OUTPUT,90900) BLK
90900    FORMAT(/'FOUND SOURCE # ',I4/)
         GENLOAD(I,J) = 1.0
      ELSE
      ENDIF
9971  CONTINUE
C
9972     TBLK = BLK
         IND  = STAK(SP)
         IEND = STAK(SP-1)
         IBEG = STAK(SP-2)
         BLK  = STAK(SP-3)
         SP   = SP - 4
C
C-----   INCREMENT INDEX OF ARC'S TAIL
C
 9990    IND = IND + 1
         GOTO 9930
 9999    CONTINUE
C
 9973 CONTINUE
C
C
C -----     BUILD AN ARRAY TO FIND THE LOAD LEVEL FOR EACH SOURCE
C
      DO 1100 I = 1, NUMSRC
        TEMP = 0.0
        DO 1110 J = 1, NUMLD
          SRCLD(I) = TEMP + GENLOAD(I,J) * EQDATA(LOAD(J),4)
          TEMP = SRCLD(I)
 1110   CONTINUE
 1100 CONTINUE
C
C -----     BUILD AN ARRAY TO CHECK IF ANY OF THE GENERATORS ARE
C -----     PARALLELED
C
      DO 1300 J = 1, NUMLD
        TEMP = 0.0
        DO 1310 I = 1, NUMSRC
          PLLCHK(J) = TEMP + GENLOAD(I,J)
          TEMP = PLLCHK(J)
 1310   CONTINUE
 1300 CONTINUE
C
      WRITE(OUTPUT, 1500)
 1500 FORMAT(/'GENERATOR/LOAD MATRIX: ROW = SOURCE #, COL = LOAD #'/)
```

```
      DO 1510 I = 1, NUMSRC
        WRITE(OUTPUT, 1520)  (GENLOAD(I,J),J=1,NUMLD)
 1520   FORMAT(6F7.1)
 1510 CONTINUE
C
      DO 1600 J = 1, NUMLD
        WRITE(OUTPUT, 1610) J, PLLCHK(J)
 1610   FORMAT(/'PLLCHK(',I4,') = ',F8.2)
 1600 CONTINUE
C
C -----     THIS SECTION DETERMINES THE LOADING OF EACH GENERATOR BY
C -----     CONSIDERING THE SHARED LOAD ON PARALLELED MACHINES
C
      DO 1700 J = 1, NUMLD
C
C -----     IF THE PARALLEL CHECK MATRIX HAS A 0.0 ENTRY, THEN THE
C -----     INDICATION IS THAT THE LOAD IS NOT POWERED BY ANY OF THE
C -----     SOURCES AVAILABLE, SO FIND OUT WHY.
C
       IF(PLLCHK(J) .EQ. 0.0) THEN
         GOTO 3000
C
C -----     AN ENTRY OF 1.0 INDICATES THAT THE LOAD IS POWERED BY
C -----     JUST ONE SOURCE, SO THE GENERATOR LOADING DUE TO THAT
C -----     LOAD IS OK, SO LOOP THROUGH AND CHECK OTHER LOADS.
C
       ELSE IF (PLLCHK(J) .EQ. 1.0) THEN
         GOTO 1700
C
C -----     ANY ENTRY GREATER THAN 1.0 INDICATES MULTIPLE POWER
C -----     SOURCES POWERING A LOAD.  THE NUMBER IN THE MATRIX
C -----     IS THE NUMBER OF POWER SOURCES CONNECTED IN PARALLEL.
C -----     THIS SECTION REDUCES THE LOADING ON ANY GENERATOR SO
C -----     THAT THE SHARED LOAD IS REFLECTED IN THE LOADING.
C
       ELSE
         DO 2100 I = 1, NUMSRC
           SRCLD(I)=SRCLD(I)-GENLOAD(I,J)*(EQDATA(LOAD(J),4)-
     &               EQDATA(LOAD(J),4)/PLLCHK(J))
 2100    CONTINUE
         GOTO 1700
       ENDIF
C
 3000 IF ( UP( LOAD(J) ) .EQ. 0 ) THEN
        LOADWN( J ) = LOAD(J)
        WRITE(OUTPUT, 3900) J, LOAD(J)
 3900   FORMAT(/'LOAD ',I4,', EQUIPMENT # ',I4,' NOT POWERED.')
      ELSE IF (HITDAM(LOAD(J)) .EQ. 1) THEN
        WRITE(OUTPUT, 3800) J, LOAD(J)
 3800   FORMAT(/'LOAD ',I4,', EQUIPMENT # ',I4,' DAMAGED.')
      ELSE
        WRITE(OUTPUT, 3850) J, LOAD(J)
 3850   FORMAT(/'LOAD ',I4,', EQUIPMENT # ',I4,' SWITCHED OFF.')
      ENDIF
C
 1700 CONTINUE
C
C -----     NOW CHECK THE SOURCES FOR OVERLOAD
```

```
C
      DO 1400 I = 1, NUMSRC
C
C -----     IF THE SOURCE POWER AVAILABLE IS LESS THAN THE LOADING,
C -----     THEN THE GENERATOR IS OK.
C
        IF (EQDATA(SOURCE(I),4) .GE. SRCLD(I)) THEN
          GOTO 1420
C
C -----     IF THE GENERATOR IS OVERLOADED, THEN MARK THE OVERLOAD LEVEL
C
        ELSE
          OVRLD(I) =  SRCLD(I) - EQDATA(SOURCE(I), 4)
          WRITE(OUTPUT, 1430) I, OVRLD(I)
 1430     FORMAT(/'SOURCE(',I4,') IS OVERLOADED BY ',F8.2,' KW.')
        ENDIF
C
 1420   WRITE(OUTPUT, 1410) I, SRCLD(I)
 1410   FORMAT(/'SRCLD(',I4,') = ',F8.2)
 1400 CONTINUE
C
      RETURN
C
      END
C
```

```
C ***************************************************************************
C ***************************************************************************
C ***************************************************************************
C
      SUBROUTINE CLOSEF
C                                                                  03/07/88
C-----------------------------------------------------------------------
C    PURPOSE:   CLOSES INPUT AND OUTPUT FILES
C    CALLED IN: MAIN
C-----------------------------------------------------------------------
C
      INTEGER    AUX1,  AUX2,  AUX3,CONSOL,OUTPUT
      COMMON / UNIT /  INPUT,OUTPUT,CONSOL,  AUX1,  AUX2,  AUX3
C
C-----------------------------------------------------------------------
C
      CLOSE ( UNIT =  INPUT )
      CLOSE ( UNIT = OUTPUT )
C      CLOSE ( UNIT =   AUX1 )
C      CLOSE ( UNIT =   AUX2 )
C      CLOSE ( UNIT =   AUX3 )
C
      RETURN
      END
C
```

```
C ***********************************************************************
C ***********************************************************************
C ***********************************************************************
C
      SUBROUTINE CSTAT (BLK, RF)
C                                                               4/8/92
C-----------------------------------------------------------------------
C    PURPOSE:   EVALUATES THE CHANGE IN THE STRUCTURE. THE
C               STRUCTURE IS REPRESENTED AS A NETWORK, PARTS
C               BEING THE SOURCES AND THE SYSTEM NODE BEING THE SINK.
C               AFTER A CONTROL ACTION TO THE BLOCK POINTED TO BY "BLK"
C               THIS ROUTINE IS CALLED. THE EFFECT OF THE CHANGE IS
C               TRACED THROUGH THE NETWORK.
C    CALLED IN: MAIN
C    CALLS TO:  ERRDKE
C    NOTE:      IN THIS SUBROUTINE THE STACK POINTER IS NOW TRACING
C               FROM THE BOTTOM UP.  IF A PIECE OF EQUIPMENT IS SHARED
C               BY SEVERAL GROUPS (RESOURCE SHARING), THE POINTER
C               PROVIDES BOTH HORIZONTAL AND VERTICAL TRACING.
C               IT STARTS AT THE EQUIPMENT LEVEL, MOVES TO THE HIGHEST
C               VERTICAL LEVEL AFFECTED BY THE CURRENT FAILURE OR REPAIR
C               EVENT, THEN MOVES HORIZONTALLY TO THE NEXT GROUP AFFECTED.
C               WHEN THERE IS NO BOTTOM-UP RESOURCE SHARING, THE STACK
C               POINTER IS SUPERFLUOUS IN THIS SUBROUTINE.
C-----------------------------------------------------------------------
C
C-----    FILE ARRAYDIM.INC CONTAINS THE MAXIMUM ARRAY DIMENSIONS AND
C         FILE DECLARE.INC  CONTAINS GLOBAL DECLARATIONS
C
          INCLUDE ARRAYDIM.INC
          INCLUDE DECLARE.INC
C
C-----    LOCAL DECLARATIONS
C
          INTEGER      RF,    SP
C
C         END OF LOCAL DECLARATIONS
C
C-----------------------------------------------------------------------
C
C-----    INITIALIZE THE STACK POINTER
C
          SP = 0
C
C-----    EVENT TYPE: RF = 0/1 -> SWITCH ON/ SWITCH OFF BLOCK
C
 1000     IF ( RF.EQ.0 ) GOTO 1100
C
C-----    SWITCH BLOCK OFF
C
          UP(BLK)      = 1
C
          GOTO 1200
C
C-----    SWITCH BLOCK ON
C
 1100     UP(BLK)      = 0
C
```

```
C-----  START TRACING THE EFFECT OF THE EVENT INVOLVING THIS
C       BLOCK ON THE REST OF THE SYSTEM
C
C-----  SET ARC INDICES
C
 1200   IBEG   = FSTAR(BLK)
        IEND   = FSTAR(BLK+1) - 1
        IND    = IBEG
C
 1300   IF ( IND .GT. IEND ) GOTO 1600
        TBLK   = ARC(IND)
        TBLKGR = TBLK - LOWGRP + 1
        IF ( RF .EQ. 0)    GOTO 1400
C
        NOUP(TBLKGR) = NOUP(TBLKGR) - 1
        IF (   UP(TBLK) .EQ. 1 )                GOTO 1700
        IF ( NOUP(TBLKGR) .GE. NONEED(TBLKGR) ) GOTO 1700
        GOTO 1500
C
 1400   NOUP(TBLKGR) = NOUP(TBLKGR) + 1
        IF (   UP(TBLK) .EQ. 0 )                GOTO 1700
        IF ( NOUP(TBLKGR) .LT. NONEED(TBLKGR) ) GOTO 1700
C
C----   THIS CODE PERFORMS THE CONVERSION WITHOUT RESETTING DWNTME.
C
 1500   SP = SP + 4
        IF ( SP.GT.LEN12 ) CALL ERRKDE (8,SP,LEN12)
        STAK(SP-3) = BLK
        STAK(SP-2) = IBEG
        STAK(SP-1) = IEND
        STAK(SP)   = IND
        BLK = TBLK
        GOTO 1000
C
 1600   IF ( SP .EQ. 0 ) RETURN
C
        TBLK = BLK
        IND  = STAK(SP)
        IEND = STAK(SP-1)
        IBEG = STAK(SP-2)
        BLK  = STAK(SP-3)
        SP   = SP - 4
C
 1700   IND = IND + 1
        GOTO 1300
C
C
        END
C
```

```
C ***********************************************************************
C ***********************************************************************
C ***********************************************************************
C
      SUBROUTINE DAMAGE
C
C                                                                 5/6/92
C-----------------------------------------------------------------------
C    PURPOSE:   CONTROLS COMPONENT DAMAGE AND EVALUATION
C    CALLED IN: MAIN
C    CALLS TO:
C-----------------------------------------------------------------------
C
C-----    FILE ARRAYDIM.INC CONTAINS THE MAXIMUM ARRAY DIMENSIONS AND
C         FILE DECLARE.INC  CONTAINS GLOBAL DECLARATIONS
C
         INCLUDE ARRAYDIM.INC
         INCLUDE DECLARE.INC
C
C-----------------------------------------------------------------------
C
C -----       LOCAL DECLARATIONS
C -----       NUMber DAMaged
C
      INTEGER      I, NUMDAM
C
C -----       END OF LOCAL DECLARATIONS
C
C -----       INITIALIZE
C
      I      = 0
      NUMDAM       = 0
C
C -----       DETERMINE THE RADIUS OF THE DAMAGE EXTENT.
C -----       THE RANDOM NUMBER GENERATOR USES SEED3.  RMAX IS THE MAXIMUM
C -----       RADIUS WHICH IS CONSIDERED FOR THE EVENT.  IT IS USER INPUT
C -----       FROM THE INPUT FILE.  IT IS USUALLY SOME PERCENTAGE OF THE
C -----       LBP OF THE SHIP SUCH AS 15% (WHICH IS THE DAMAGE LENGTH
C -----       CRITERIA FOR SHIP DESIGN).  THIS ROUTINE IS SIMPLE NOW, BUT
C -----       IF IT IS DETERMINED THAT A MORE COMPLETE DAMAGE CALCULATION
C -----       IS TO BE USED, THIS CAN BE EXPANDED.
C
C -----       THIS ORIGINAL
C -----       VERSION ASSUMES UNIFORMLY DISTRIBUTED DAMAGE RADII.
C
      RMIN  = HITDATA(4)
      RMAX  = HITDATA(5)
10    R     = RAND(SEED3)
      RDAM  = RMIN + R * (RMAX - RMIN)
C
      WRITE(OUTPUT, 90802) RDAM
      WRITE(CONSOL, 90802) RDAM
90802 FORMAT(/'THE DAMAGE RADIUS IS ',F8.2,' FEET.'/)
C
C
C -----       THIS ROUTINE COMPARES THE RADIUS OF EACH
C -----       COMPONENT TO THE RADIUS TO THE HIT, AND IF WITHIN THE HIT
C -----       REGION, SET THE UP FLAG TO 1 (DAMAGED) IF THE SHIELDing VALUE
```

```
C -----     IS EXCEEDED.  HiT MAGnitude 0 (zero) IS THE MAXIMUM MAGNITUDE
C -----     OF THE BURST AS INPUT BY THE USER.  Decay Factor IS USED TO
C -----     ALLOW A VARIABILITY IN THE BLAST MAGNITUDE WITH DISTANCE.
C
      HTMAG0 = HITDATA(6)
      DF = HITDATA(7)
C
      DO 200 I = 1, NPARTS
C
      IF ( RADIUS(I) .GT. RDAM ) GOTO 200
C
C -----     SHIELDing VALUE FOR THE COMPONENT IS AS INPUT.  THE HiT
C -----     MAGnitude IS FOUND AT EACH RADIUS FROM THE HIT TO THE
C -----     COMPONENT.  THE SHIELDING VALUE IS COMPARED TO THE HIT
C -----     MAGNITUDE TO DETERMINE IF THE COMPONENT WILL BE PUT DOWN.
C
      SHIELD = EQDATA(I, 7)
      HTMAG = HTMAG0 * EXP( (-1) * DF * RADIUS (I) )
      IF (HTMAG .LE. SHIELD) GOTO 200
C
C -----     IF THE COMPONENT IS HIT DAMaged, MARK IT AS SUCH TO KEEP
C -----     TRACK OF THE COMPONENTS WHICH ARE DOWN DUE TO HIT AND WHICH
C -----     ARE DOWN DUE TO CONNECTION COMPONENT DAMAGE.
C
      HITDAM(I) = 1
      CALL FAIL(I)
      CALL STAT(I,1)
      CALL SYS_STATS
 200  CONTINUE
C
      RETURN
C
      END
C
C **********************************************************************
C **********************************************************************
C **********************************************************************
C
         SUBROUTINE ERRKDE (KODE,NUM,ITEST)
C                                                                5/5/92
C-----------------------------------------------------------------------
C    PURPOSE:   PRINTS OUT ERROR MESSAGES AS DETECTED.
C    CALLED IN: BAKST1,BAKST2,BAKST3,FILE,READIT,REMOVE,STAT.
C-----------------------------------------------------------------------
C
         INTEGER    AUX1,  AUX2,  AUX3,CONSOL,OUTPUT
         COMMON / UNIT /  INPUT,OUTPUT,CONSOL,  AUX1,  AUX2,  AUX3
C
C-----------------------------------------------------------------------
C
         WRITE (OUTPUT,100) KODE
  100    FORMAT(' *** BEAVER RUNTIME ERROR NO: ',I3,' DETECTED ***')
C
         GOTO (1,2,3,4,5,6,7,8,9,10,11,12,13,14,15), KODE
C
    1    WRITE (OUTPUT,101) ITEST
         WRITE (CONSOL,101) ITEST
  101    FORMAT(' *** MORE THAN ',I6,' VDC EQUIPMENTS ***')
```

```
      GOTO 999
C
    2 WRITE (OUTPUT,102) NUM,ITEST
      WRITE (CONSOL,102) NUM,ITEST
  102 FORMAT(' *** EQUIPMENT TYPE NO. ',I6,' > ',I6,/,
     & ' *** IN EQUIPMENT TYPE DEFINITION')
      GOTO 999
C
    3 WRITE (OUTPUT,103) NUM,ITEST
      WRITE (CONSOL,103) NUM,ITEST
  103 FORMAT(' *** EQUIPMENT TYPE NO. ',I6,' > ',I6,/,
     & ' *** IN EQUIPMENT TYPE/EQUIPMENT NUMBER DEFINITION')
      GOTO 999
C
    4 WRITE (OUTPUT,104) NUM,ITEST
      WRITE (CONSOL,104) NUM,ITEST
  104 FORMAT(' *** EQUIPMENT NO. ',I6,' > ',I6,/,
     & ' *** IN EQUIPMENT TYPE/EQUIPMENT NUMBER DEFINITION')
      GOTO 999
C
    5 WRITE (OUTPUT,105) NUM,ITEST
      WRITE (CONSOL,105) NUM,ITEST
  105 FORMAT(' *** GROUP NO. ',I7,' > ',I6,' ***')
      GOTO 999
C
    6 WRITE (OUTPUT,106) ITEST
      WRITE (CONSOL,106) ITEST
  106 FORMAT(' *** NO. OF ARCS > ',I6,' (REDIMENSION) ***')
      GOTO 999
C
    7 WRITE (OUTPUT,107) NUM
      WRITE (CONSOL,107) NUM
  107 FORMAT(' *** SUBSYSTEM NO. ',I6,' NOT CONNECTED TO SYSTEM ***')
      GOTO 999
C
    8 WRITE (OUTPUT,108)
      WRITE (CONSOL,108)
  108 FORMAT(' *** STACK TOO LARGE IN SUBROUTINE STAT ***')
      GOTO 999
C
    9 WRITE (OUTPUT,109)
      WRITE (CONSOL,109)
  109 FORMAT(' *** STACK TOO LARGE IN SUBROUTINE BAKST1 ***')
      GOTO 999
C
   10 WRITE (OUTPUT,110)
      WRITE (CONSOL,110)
  110 FORMAT(' *** STACK TOO LARGE IN SUBROUTINE BAKST2 ***')
      GOTO 999
C
   11 WRITE (OUTPUT,111)
      WRITE (CONSOL,111)
  111 FORMAT(' *** STACK TOO LARGE IN SUBROUTINE BAKST3 ***')
      GOTO 999
C
   12 WRITE (OUTPUT,112)
      WRITE (CONSOL,112)
  112 FORMAT(' *** STACK TOO LARGE IN SUBROUTINE GSTAT ***')
```

```
      GOTO 999
C
   13 WRITE (OUTPUT,113) NUM,ITEST
      WRITE (CONSOL,113) NUM,ITEST
  113 FORMAT(' *** GROUP NO. ',I7,' < ',I6,' ***')
      GOTO 999
C
   14 WRITE (OUTPUT,114) NUM
      WRITE (CONSOL,114) NUM
  114 FORMAT(' *** SERVICE QUEUE CAPACITY OF ',I6,' EXCEEDED ***')
      GOTO 999
C
   15 WRITE (OUTPUT,115)
      WRITE (CONSOL,115)
  115 FORMAT(' *** ATTEMPT TO REMOVE ITEM FROM EMPTY QUEUE ***')
C
  999 WRITE (CONSOL,1000)
      WRITE (CONSOL,1010) KODE
      WRITE (CONSOL,1000)
      WRITE (CONSOL,1020)
      PAUSE
C
      CALL CLOSEF
      STOP
C
 1000 FORMAT(' -------------------------------------------- ')
 1010 FORMAT('*** BEAVER: DAM!, RUNTIME ERROR NO: ',I3,' DETECTED ***')
 1020 FORMAT(' < HIT RETURN TO CONTINUE > ')

      END
C
C **********************************************************************
C **********************************************************************
C **********************************************************************
C
      SUBROUTINE FAIL (BLK)
C                                                               02/29/88
C----------------------------------------------------------------------
C   PURPOSE:   HANDLES FAILED PARTS, THEIR REPAIR OR FILING
C   CALLED IN: MAIN
C----------------------------------------------------------------------
C
C-----  FILE ARRAYDIM.INC CONTAINS THE ARRAY DIMENSIONS AND
C       FILE DECLARE.INC CONTAINS GLOBAL DECLARATIONS
C
      INCLUDE ARRAYDIM.INC
      INCLUDE DECLARE.INC
C
C----------------------------------------------------------------------
C
C-----  UPDATE FAILURE COUNT AND EVENT RATE
C
      NTFALS = NTFALS + 1
      RATE   = RATE - LAM(TYPE(BLK))
C
C-----  SPARES EXHAUSTED - INCREMENT NUMBER OF OUTAGES
C
      IF ( SPARES(TYPE(BLK)) .GT. 0 ) GOTO 500
```

```
      NOSPRE(TYPE(BLK)) = NOSPRE(TYPE(BLK)) + 1
      RETURN
C
C-----  SPARES AVAILABLE - FOUR POSSIBILITIES:
C
C-----  CASE 1: SERVER NOT WORKING DURING THIS PHASE -> FILE PART
C
  500   IF ( SRVOP .NE. 0 ) GOTO 1000
C
C-----  CASE 2: SERVER NOT IDLE NOW -> FILE PART
C
        IF ( REMSTG .GT.0 ) GOTO 1000
C
C-----  CASE 3: PART NOT REPAIRABLE DURING THIS PHASE -> FILE IT
C
        IF ( MU(TYPE(BLK),PHASE) .LE. 0.0 ) GOTO 1000
C
C-----  CASE 4: BEGIN SERVICE ON THE PART; ADJUST REPAIR RATE;
C-----          NOTICE THAT THE PART IS NOT SPECIFICALLY PLACED
C               IN A DESIGNATED POSITION
C
        SRVMU  = MU(TYPE(BLK),PHASE)
        RATE   = RATE + SRVMU * NSTAG
        REMSTG = NSTAG
        SRVBLK = BLK
        RTIME  = TIME
        RETURN
C
C-----  FILE PART WAITING FOR REPAIR IN THE QUEUE
C
 1000   IPOINT = FQFRST
C
        CALL REMOVE (FQ,IPOINT)
C
        PRT(IPOINT) = BLK
        RMS(IPOINT) = NSTAG
C
        CALL FILE (SQ,IPOINT,3)
        RETURN
C
        END
C
```

```
C ***********************************************************************
C ***********************************************************************
C ***********************************************************************
C
      SUBROUTINE FILE (Q,REC,OPT)
C                                                              02/26/88
C-----------------------------------------------------------------------
C    PURPOSE:   INSERTS THE RECORD POINTED TO BY THE VARIABLE "REC"
C               IN THE LIST DESCRIBED IN THE VECTOR "Q" ACCORDING
C               TO THE OPTION "OPT":
C                    1 - FILE IN LAST PLACE
C                    2 - FILE IN FIRST PLACE
C                    3 - FILE IN ASCENDING PRIORITY, LAST WITHIN CLASS
C                    4 - FILE IN ASCENDING PRIORITY, FIRST WITHIN CLASS
C    CALLED IN: FAIL, PCHNG, REPAIR
C    CALLS TO:  ERRKDE
C    NOTE 1:    THIS SUBROUTINE WAS ALTERED WHEN ASKING ABOUT THE
C               REPAIR PRIORITY OF FAILED COMPONENTS (12/10/87).
C    NOTE 2:    A LOWER VALUED PRIORITY MEANS A MORE CRITICAL EQUIPMENT.
C    NOTE 3:    IN OPT 3 EQUIPMENTS ARE FILED IN ASCENDING ORDER OF
C               PRIORITY AND LAST WITHIN A PRIORITY CLASS. IN THIS
C               OPTION AN EQUIPMENT IS ADVANCED FROM THE BACK FORWARD
C               AS LONG AS ITS PRIORITY IS LESS THAN THAT OF THE ELEMENT
C               IT IS COMPARED WITH. THIS PROCEDURE INSURES THAT THE
C               ELEMENT WILL BE FILED LAST WITHIN ITS OWN PRIORITY CLASS.
C    NOTE 4:    IN OPT 4 AN EQUIPMENT IS STILL FILED IN ASCENDING
C               PRIORITY BUT IS PLACED FIRST WITHIN ITS CLASS. THUS THE
C               EQUIPMENT IS STARTED AT THE FRONT AND ADVANCED TOWARDS
C               THE BACK OF THE QUEUE. THIS INSURES THAT THE EQUIPMENT
C               WILL BE PLACED FIRST WITHIN ITS OWN PRIORITY CLASS.
C
C-----------------------------------------------------------------------
C
C-----    FILE ARRAYDIM.INC CONTAINS THE ARRAY DIMENSIONS AND
C         FILE DECLARE.INC CONTAINS GLOBAL DECLARATIONS
C
      INCLUDE ARRAYDIM.INC
      INCLUDE DECLARE.INC
C
C-----    LOCAL DECLARATIONS
C
      INTEGER  BPOINT, OPT, POINT, REC, Q
      DIMENSION Q(3)
C
C-----------------------------------------------------------------------
C
C-----    INCREMENT LIST COUNT
C
      NINLST = NINLST + 1
      IF ( NINLST .GT. MAXLST ) THEN
      WRITE(OUTPUT,25)
      WRITE(CONSOL,25)
   25    FORMAT(/' ERROR FOUND IN  SUBROUTINE FILE')
      CALL ERRKDE(14,NINLST,0)
      ENDIF
      IF ( Q(3) .GT. 0 ) GOTO 50
C
C-----    THE FIRST ELEMENT TO ENTER THE QUEUE
```

```
C
      Q(1) = REC
      Q(2) = REC
      Q(3) = 1
      SUCC(REC) = 0
      PRED(REC) = 0
      GOTO 9999
C
   50 GOTO (1000,2000,3000,4000),OPT
C                 ^      ^      ^      ^
C-----          Q(2)  Q(1)  Q(2)  Q(1)
C
C-----    FILE THE ELEMENT IN LAST PLACE
C
 1000 PRED(REC) = Q(2)
      SUCC(REC) = 0
      SUCC(Q(2)) = REC
      Q(2) = REC
      GOTO 5000
C
C-----    FI-E THE ELEMENT IN FIRST PLACE
C
 2000 PRED(REC) = 0
      SUCC(REC) = Q(1)
      PRED(Q(1)) = REC
      Q(1) = REC
      GOTO 5000
C
C-----    FILE THE ELEMENT IN ASCENDING PRIORITY, LAST WITHIN CLASS
C
 3000 IF ( PRI(TYPE(PRT(REC))) .GE. PRI(TYPE(PRT(Q(2)))) )
     &    GOTO 1000
      POINT = Q(2)
C
C-----    ADVANCING FROM THE BACK FORWARD!!!!
C
 3100 POINT = PRED(POINT)
C
C-----    WHEN POINT = 0 YOU HAVE GONE BEYOND THE BOUNDS OF THE QUEUE.
C
      IF ( POINT .EQ. 0 ) GOTO 2000
      IF ( PRI(TYPE(PRT(REC))) .LT. PRI(TYPE(PRT(POINT))) )
     &    GOTO 3100
C
C-----    HERE IS WHERE REC IS INSERTED
C
      BPOINT = SUCC(POINT)
      SUCC(REC) = BPOINT
      PRED(REC) = POINT
      SUCC(POINT)  = REC
      PRED(BPOINT) = REC
      GOTO 5000
C
C-----    FILE THE ELEMENT IN ASCENDING PRIORITY, FIRST WITHIN CLASS
C
 4000 IF ( PRI(TYPE(PRT(REC))) .LE. PRI(TYPE(PRT(Q(1)))) )
     &    GOTO 2000
      POINT = Q(1)
```

```
C
C-----    ADVANCING FROM FRONT TO BACK!!!!
C
 4100     POINT = SUCC(POINT)
C
C-----    WHEN POINT = 0 YOU HAVE GONE BEYOND THE BOUNDS OF THE QUEUE.
C
          IF ( POINT .EQ. 0 ) GOTO 1000
          IF ( PRI(TYPE(PRT(REC))) .GT. PRI(TYPE(PRT(POINT))) )
     &       GOTO 4100
C
C-----    HERE IS WHERE REC IS INSERTED
C
          BPOINT = PRED(POINT)
          SUCC(REC) = POINT
          PRED(REC) = BPOINT
          SUCC(BPOINT) = REC
          PRED(POINT)  = REC
 5000     Q(3) = Q(3) + 1
C
 9999     RETURN
          END
C
```

```
C ************************************************************************
C ************************************************************************
C ************************************************************************
C
      SUBROUTINE GENVNT (EVENT, BLK)
C                                                                 10/03/88
C-----------------------------------------------------------------------
C   PURPOSE:   UPDATES THE PROBABILITY DISTRIBUTION FUNCTION AND THE
C              ASSOCIATED EVENT RATE; GENERATES THE NEXT EVENT TIME,
C              THE EQUIPMENT INVOLVED AND THE TYPE OF EVENT:
C                   1 - FAILURE
C                   2 - REPAIR
C                   3 - PHASE CHANGE
C   CALLED IN: MAIN
C   CALLS TO:  RAND
C-----------------------------------------------------------------------
C
C-----      FILE ARRAYDIM.INC CONTAINS THE ARRAY DIMENSIONS AND
C           FILE DECLARE.INC CONTAINS GLOBAL DECLARATIONS
C
      INCLUDE ARRAYDIM.INC
      INCLUDE DECLARE.INC
C
C-----      LOCAL DECLARATIONS
C
      INTEGER   EVENT
C
      REAL*8    RATELO
C
      DATA  RATELO /1.0D-8/
C
C-----      END OF LOCAL DECLARATIONS
C
C-----------------------------------------------------------------------
C
C-----      UPDATE PROBABILITY DISTRIBUTION FUNCTION AND EVENT RATE
C
      P(1)    = 0.0
      DO 20 I = 1,NPARTS
         P(I+1) = P(I)
         IF ( UP(I) .EQ. 0 ) P(I+1) = P(I+1) + LAM(TYPE(I))
   20 CONTINUE
      P(NP2) = RATE
C
C-----      TEST FOR A TOO LOW EVENT RATE
C
      IF ( RATE .GE. RATELO ) GOTO 40
      WRITE (CONSOL,9000) RATE,TIME
      GOTO 60
C
C-----      ADVANCE SIMULATION CLOCK ASSUMING AN EXPONENTIAL RELATIONSHIP,
C           I.E., R = EXP(-RATE*TIME)
C
   40 R = RAND(SEED1)
      TIME = - ALOG(R) / RATE + TIME
      IF ( TIME .LT. PHTIME(PHI) ) GOTO 80
C
C-----      EVENT IS PHASE CHANGE IF EITHER RATE IS TOO LOW OR
```

```
C        PREDICTED TIME EXCEEDS NEXT PHASE CHANGE BOUNDARY
C
   60    TIME = PHTIME(PHI)
         EVENT = 3
         GOTO 1000
C
C-----   FIND EQUIPMENT INVOLVED (IF ANY) THROUGH A BINARY SEARCH
C        OVER THE PART FAILURE PROBABILITY DISTRIBUTION; COMPARING
C        P(M) TO Z IS THE ESSENCE OF SAMPLING FOR THE FAILED PART
C        OR DETERMINING A REPAIR EVENT.
C
   80    R = RAND(SEED2)
         Z = R * P(NP2)
C
         MLO = 1
         MHI = NP2
  100    M = ( MLO + MHI ) / 2
         IF ( MHI - MLO .LE. 1) GOTO 400
         IF ( P(M) - Z ) 200,500,300
C
  200    MLO = M
         GOTO 100
C
  300    MHI = M
         GOTO 100
C
  400    M = MLO
  500    CONTINUE
C
C-----   EVENT IS REPAIR IF M>NPARTS
C
         IF ( M .LE. NPARTS) GO TO 600
         BLK = SRVBLK
         EVENT = 2
         GOTO 1000
C
C-----   EVENT IS FAILURE, PROVIDED THE EQUIPMENT IS OPERATIONAL
C
  600    CONTINUE
C
C-----   CHECK IF EQUIPMENT IS OF VARIABLE DUTY CYCLE TYPE
C
         IVDCTM = IVDC(TYPE(M))
         IF ( IVDCTM .EQ. 0 ) GOTO 900
C
C-----   IF EQUIPMENT IS DORMANT, RESAMPLE WITH TIME ADVANCE
C
         R = RAND(SEED3)
         IF ( R .GT. VDC(IVDCTM,PHASE) ) GOTO 40
C
  900    EVENT = 1
         BLK = M
C
 1000    RETURN
C
 9000    FORMAT(' EVENT RATE ',E10.3,' IS TOO SMALL AT TIME = ',F10.3)
C
         END
```

```
C ************************************************************************
C ************************************************************************
C ************************************************************************
C
      SUBROUTINE HEADER
C                                                                 5/6/92
C------------------------------------------------------------------------
C    PURPOSE:   PRINTS PROGRAM IDENTIFICATION HEADER
C    CALLED IN: MAIN
C------------------------------------------------------------------------
C
      INTEGER    AUX1,  AUX2,  AUX3,CONSOL,OUTPUT
      COMMON / UNIT /  INPUT,OUTPUT,CONSOL,  AUX1,  AUX2,  AUX3
C
C----     LOCAL VARIABLES FOR THE TIME AND DATE STAMP
C
      INTEGER HOUR, MIN, SECONDS, MM, DD, YY
C
C----     END OF LOCAL VARIABLES
C
C------------------------------------------------------------------------
C
      WRITE (OUTPUT,90020)
90020 FORMAT(//
     1,8X,'===============================================================',/
     2,8X,'==        BEAVER   WITH SURVIVABILITY ANALYSIS                ==',/
     3,8X,'==                   MAY 7, 1992                              ==',/
     4,8X,'==        CLIFF WHITCOMB  (FOR MIT THESIS) WITH               ==',/
     5,8X,'==          CHARLES STARK DRAPER LABORATORY                   ==',/
     6,8X,'==            SYSTEM SCIENCES DIVISION                        ==',/
     7,8X,'==          CAMBRIDGE, MASSACHUSETTS 02139                    ==',/
     8,8X,'===============================================================')
C
C---- PRINT OUT THE DAY AND TIME THIS FILE WAS RUN
C
      CALL DATE (MM,DD,YY)
      CALL TIME (SECONDS)
C
      HOUR    = SECONDS / 3600
      DUMMY   = HOUR * 3600
      SECONDS = SECONDS - DUMMY
      MIN     = SECONDS / 60
      DUMMY   = MIN * 60
      SECONDS = SECONDS - DUMMY
C
      WRITE (OUTPUT,90300) MM, DD, YY, HOUR, MIN, SECONDS
90300 FORMAT(/,' *****    BEAVER TALE WAS CREATED ON ',
     &           I2.2,2('/',I2.2),' AT ',I2.2,2(':',I2.2),'     *****')
C
      RETURN
      END
C
```

```
C ***************************************************************************
C ***************************************************************************
C ***************************************************************************
C
      SUBROUTINE HSORT (B,N)
C                                                                   02/29/88
C---------------------------------------------------------------------------
C    PURPOSE:    SORTS LIST B IN ASCENDING ORDER
C    CALLED IN: MAIN
C    NOTE:        "IT" IS INTEGER AND USED AS TEMPORARY
C                 STORAGE FOR B(.) WHICH IS FLOATING POINT!
C---------------------------------------------------------------------------
C
      DIMENSION  B(N)
C
C---------------------------------------------------------------------------
C
      DO 50 I = 2,N
         J = I
 25      K = J / 2
         IF ( B(J) .LT. B(K) ) GOTO 50
         IT   = B(K)
         B(K) = B(J)
         B(J) = IT
         J = K
         IF ( J .GT. 1 ) GOTO 25
 50   CONTINUE
C
      I = N
100   IF ( I .LT. 1 ) RETURN
      IT   = B(1)
      B(1) = B(I)
      B(I) = IT
      I = I - 1
      J = 1
      K = 2 * J
110   IF ( K .GT. I ) GOTO 100
         IF ( K .LT. I .AND. B(K) .LT. B(K+1) ) K = K + 1
         IF ( B(J) .GT. B(K) ) GOTO 100
         IT   = B(J)
         B(J) = B(K)
         B(K) = IT
         J = K
         K = 2 * J
      GOTO 110
C
      END
C
```

```
C ***********************************************************************
C ***********************************************************************
C ***********************************************************************
C
      SUBROUTINE INITAL
C                                                               02/29/88
C-----------------------------------------------------------------------
C    PURPOSE:   INITIALIZES LINKED LISTS AND STATISTICS
C               FOR EACH MISSION.
C    CALLED IN: MAIN
C    CALLS TO:  SET
C-----------------------------------------------------------------------
C
C-----    FILE ARRAYDIM.INC CONTAINS THE ARRAY DIMENSIONS AND
C         FILE DECLARE.INC CONTAINS GLOBAL DECLARATIONS
C
      INCLUDE ARRAYDIM.INC
      INCLUDE DECLARE.INC
C
C-----------------------------------------------------------------------
C
      NINLST = 0
      PHI    = 1
      PHASE  = PHZ(PHI)
      SRVOP  = PHSRV(PHASE)
      REMSTG = 0
      SRVBLK = 0
      RTIME  = 0.0
      TIME   = 0.0
      DONE   = .FALSE.
C
      FQFRST = 1
      FQLAST = MAXLST
      FQNUM  = MAXLST
      SQFRST = 0
      SQLAST = 0
      SQNUM  = 0
C
C-----    INITIALIZE AVAILABLE SPARES BY EQUIPMENT TYPE
C
      DO 1000 I = 1,NEQT
         SPARES(I) = SPARE1(I)
 1000 CONTINUE
C
C-----    INITIALIZES VARIABLES RELEVANT TO BOTH PARTS AND GROUPS
C
      DO 2000 I = 1,NBLKS
         CDWNTM(I) = 0.0
         DWNTME(I) = 0.0
         NUMFAL(I) = 0
         SYSDWN(I) = 0
         UP(I)     = 0
 2000 CONTINUE
C
C-----    INITIALIZES VARIABLES RELEVANT ONLY TO GROUPS OF EQUIPMENTS
C
      DO 2500 I = 1,NGROUP
```

```
            NOUP(I)     = NOOFF(I)
 2500   CONTINUE
C
C-----  INITIALIZE LIST VARIABLES
C
        DO 3000 I = 2,MAXLST
           SUCC(I-1) = I
           PRED(I) = I-1
 3000   CONTINUE
C
        SUCC(MAXLST) = 0
        PRED(1)      = 0
C
C-----  INITIALIZE PROBABILITY DISTRIBUTION FUNCTION AND EVENT RATE
C
        P(1)    = 0.0
        DO 4000 I = 1,NPARTS
           P(I+1) = P(I) + LAM(TYPE(I))
 4000   CONTINUE
        P(NP2) = P(NP1)
        RATE   = P(NP2)
C
C-----  VARIABLES IN THE INITIAL PHASE ARE INITIALIZED IN SET
C
        CALL SET
C
        RETURN
        END
C
```

```
C ************************************************************************
C ************************************************************************
C ************************************************************************
C
         SUBROUTINE LCABLE(BLK)
C                                                                 5/6/92
C-----------------------------------------------------------------------
C     PURPOSE:    DETERMINES CABLE LENGTH FOR COST ESTIMATE.
C     CALLED IN:  READIT
C     CALLS TO:
C-----------------------------------------------------------------------
C
C-----    FILE ARRAYDIM.INC CONTAINS THE MAXIMUM ARRAY DIMENSIONS AND
C         FILE DECLARE.INC  CONTAINS GLOBAL DECLARATIONS
C
         INCLUDE ARRAYDIM.INC
         INCLUDE DECLARE.INC
C
      CX1 = EQDATA(BLK, 1)
      CY1 = EQDATA(BLK, 2)
      CZ1 = EQDATA(BLK, 3)
      CX2 = EQDATA(BLK, 4)
      CY2 = EQDATA(BLK, 5)
      CZ2 = EQDATA(BLK, 6)
C
      AA = CX2 - CX1
      BB = CY2 - CY1
      CC = CZ2 - CZ1
C
      CBLENGTH = SQRT(AA**2 + BB**2 + CC**2)
C
      RETURN
C
      END
C
```

```
C ***********************************************************************
C ***********************************************************************
C ***********************************************************************
C
        SUBROUTINE LINEUP
C                                                               5/6/92
C-----------------------------------------------------------------------
C   PURPOSE:  CHECKS THE EQDATA() STATUS FIELD TO SET THE UP FLAGS
C             ACCORDING TO THE DESIRED LINE UP.  UP(EQUIPNO) IS SET
C             TO 1 IF THE DESIRED STATUS FLAG IS SET TO 1.
C             A DESIRED STATUS OF "1" IS OFF, AND OF "0" IS ON.
C   CALLED IN: MAIN
C   CALLS TO:
C-----------------------------------------------------------------------
C
C-----  FILE ARRAYDIM.INC CONTAINS THE MAXIMUM ARRAY DIMENSIONS AND
C       FILE DECLARE.INC  CONTAINS GLOBAL DECLARATIONS
C
        INCLUDE ARRAYDIM.INC
        INCLUDE DECLARE.INC
C
C-----  LOCAL DECLARATIONS
C
      INTEGER    ONOFF
C
C -----     ONOFF IS THE STATUS FIELD OF THE EQDATA ARRAY
C
      ONFLAG = 0.0
      ONOFF = 5
C
C -----     GET THE DESIRED STATUS FLAG FOR THE EQUIPMENT AND SET THE
C -----     UP FLAG EQUAL TO 1 IF DESIRED STATUS IS OFF.
C
C
      DO 100 I = 1, NPARTS
      EQCODE = EQTYPE( I )
      GOTO ( 10, 20, 30, 40, 50, 60, 70, 80 ), EQCODE
C
C -----     ALTERNATE DC POWER SOURCE
C
 10   ONFLAG = EQDATA(I,ONOFF)
      GOTO 100
C
C -----     POWER SOURCE
C
 20   ONFLAG = EQDATA(I,ONOFF)
      GOTO 90
C
C -----     LOAD
C
 30   ONFLAG = EQDATA(I,ONOFF)
      GOTO 90
C
C -----     BREAKER
C
 40   ONFLAG = EQDATA(I,ONOFF)
      GOTO 90
C
```

```
C -----      CABLE CANNOT BE CONTROLLED
C
 50    GOTO 100
C
C -----      CABINET CANNOT BE CONTROLLED
C
 60    GOTO 100
C
C -----      BUS TIE
C
 70    ONFLAG = EQDATA(I,ONOFF)
       GOTO 90
C
C -----      ABT LINE UP CONTOLLED BY THE WAY IT IS PUT IN THE INPUT FILE
C -----      NOT BY A DYNAMIC PROCESS.
C
 80    GOTO 100
C
 90    IF (ONFLAG .EQ. 0.0) GOTO 100
C
C -----      MAKE CALLS TO ADJUST SYSTEM RATE
C -----      AND SET UP(I) = 1.  ADJUST THE SYSTEM RATE SINCE THE
C -----      COMPONENT IS BEING TAKEN OUT OF SERVICE.
C
       RATE     = RATE - LAM(TYPE(I))
C
C -----      CSTAT IS THE SAME AS STAT EXCEPT THAT NO STATISTICS ARE
C -----      KEPT AS TO THE DOWN TIME OF THE COMPONENT SINCE IT IS IN
C -----      A Controlled STATus STATE.
C
       WRITE(OUTPUT, 500) I
       WRITE(CONSOL, 500) I
 500   FORMAT(/'SWITCHING EQUIPMENT # ',I4,' OFF BEFORE ANALYSIS.'/)
C
       CALL CSTAT(I, 1)
       CALL SYS_STATS
C
C
 100   CONTINUE
C
       RETURN
C
       END
C
```

```
C ***********************************************************************
C ***********************************************************************
C ***********************************************************************
C
      SUBROUTINE OPENF
C                                                                 3/20/92
C-----------------------------------------------------------------------
C    PURPOSE:   OPENS INPUT AND OUTPUT FILES - UPDATED FOR BEAVER
C    CALLED IN: MAIN
C-----------------------------------------------------------------------
C
      INTEGER    AUX1,  AUX2,  AUX3,CONSOL,OUTPUT
      COMMON / UNIT /  INPUT,OUTPUT,CONSOL,  AUX1,  AUX2,  AUX3
C
C-----------------------------------------------------------------------
C
      OPEN ( UNIT =  INPUT, FILE = 'BEAVER.IN',  STATUS = 'OLD' )
      OPEN ( UNIT = OUTPUT, FILE = 'BEAVER TALE', STATUS = 'NEW' )
C      OPEN ( UNIT =   AUX1, FILE = 'BEAVER.AX1', STATUS = 'NEW' )
C      OPEN ( UNIT =   AUX2, FILE = 'BEAVER.AX2', STATUS = 'NEW' )
C      OPEN ( UNIT =   AUX3, FILE = 'BEAVER.FTF', STATUS = 'NEW' )
C
      RETURN
      END
C
```

```
C *********************************************************************
C *********************************************************************
C *********************************************************************
C
        SUBROUTINE PCHANG
C                                                                2/29/88
C-----------------------------------------------------------------------
C     PURPOSE:   SERVICES THE CHANGE OF PHASE EVENT
C     CALLED IN: MAIN
C     CALLS TO:  FILE, REMOVE, SET
C     NOTE:      THE LAST POSITION OF THE FREE QUEUE IS DESIGNATED
C                THE IN-SERVICE POSITION.
C-----------------------------------------------------------------------
C
C-----  FILE ARRAYDIM.INC CONTAINS THE ARRAY DIMENSIONS AND
C       FILE DECLARE.INC CONTAINS GLOBAL DECLARATIONS
C
        INCLUDE ARRAYDIM.INC
        INCLUDE DECLARE.INC
C
C-----  LOCAL DECLARATIONS
C
        INTEGER POINT
C
C-----  END OF LOCAL DECLARATIONS
C
C-----------------------------------------------------------------------
C
C-----  INCREMENT THE PHASE INDEX
C
        PHI = PHI + 1
C
C-----  CHECK FOR END OF MISSION FLAG (PHZ = 0)
C
        IF ( PHZ(PHI) .NE. 0 ) GOTO 500
        DONE = .TRUE.
        GOTO 9999
C
C-----  MISSION CONTINUES
C
  500   PHASE = PHZ(PHI)
C
C-----  RECONFIGURE FOR THIS PHASE TYPE
C
        CALL SET
C
C-----  CHECK FOR SERVER STATUS CHANGE
C
        IF ( SRVOP .EQ. PHSRV(PHASE) ) GOTO 4000
C
C-----  CHANGE SERVER STATUS AND ASSESS CURRENT STATUS
C
        SRVOP = PHSRV(PHASE)
        IF ( SRVOP .NE. 0) GOTO 2000
C
C-----  TURN SERVER ON AND LOOK FOR A SERVICEABLE PART
C
        IF ( SQNUM .EQ. 0 ) GOTO 9999
```

```
      POINT = SQFRST
 1000 K = TYPE(PRT(POINT))
      IF ( MU(K,PHASE) .GT. 0.0 ) GOTO 1100
 1050 POINT = SUCC(POINT)
      IF ( POINT .NE. 0 ) GOTO 1000
      GOTO 9999
C
C-----    EQUIPMENT IS REPAIRABLE
C
 1100 CALL REMOVE(SQ,POINT)
      CALL FILE(FQ,POINT,1)
C
      IF ( SPARES(K) .GT. 0 ) GOTO 1150
C
C-----    SPARES NO LONGER AVAILABLE; INCREMENT SPARE OUTAGE COUNTER
C
      NOSPRE(K) = NOSPRE(K) + 1
      GOTO 1050
C
C-----    SPARES STILL AVAILABLE
C
 1150 SRVMU  = MU(K,PHASE)
      RATE   = RATE  +  SRVMU * NSTAG
      REMSTG = RMS(POINT)
      SRVBLK = PRT(POINT)
      RTIME  = TIME
      GOTO 9999
C
C-----    TURN SERVER OFF; ANY PART IN SERVICE IS RETURNED
C         TO THE SERVICE QUEUE FROM THE LAST POSITION OF
C         THE FREE QUEUE- THE SO-DESIGNATED IN-SERVICE POSITION.
C
 2000 IF ( REMSTG .EQ. 0) GOTO 9999
C
C-----** THE NEXT LINE IS LOGICALLY INCONSISTENT:
C     ** IT SHOULD BE POINT = FQLST !!!!
C
      POINT = FQFRST
      CALL REMOVE(FQ,POINT)
      PRT(POINT) = SRVBLK
      RMS(POINT) = REMSTG
      CALL FILE(SQ,POINT,4)
C
C-----    CLEAN OFF SERVER
C
      SRVBLK = 0
      REMSTG = 0
      RATE   = RATE - SRVMU * NSTAG
      GOTO 9999
C
C-----    SERVER STATUS REMAINS THE SAME
C
 4000 IF ( SRVOP .NE. 0 ) GOTO 9999
C
C-----    SERVER REMAINS ON
C
      IF ( REMSTG .EQ. 0 ) GOTO 4250
      IF ( MU(TYPE(SRVBLK),PHASE) .LE. 0.0 ) GOTO 4200
```

```
C
C-----   EQUIPMENT IS BEING REPAIRED; THE PROGRAM MUST ACCOUNT
C        FOR POSSIBLE CHANGE IN REPAIR RATE DUE TO THE PHASE
C        CHANGE. THIS IS ACCOUNTED FOR BY THE VARIABLE X.
C
         X      = MU(TYPE(SRVBLK),PHASE) - SRVMU
         RATE   = RATE  +  X * NSTAG
         SRVMU  = MU(TYPE(SRVBLK),PHASE)
         GOTO 9999
C
C-----   PART BECOMES NON-REPAIRABLE DUE TO PHASE CHANGE; IT IS
C        ROTATED BACK INTO THE SERVICE QUEUE FROM THE FREE QUEUE.
C
 4200    RATE   = RATE  -  SRVMU * NSTAG
C
C-----** THIS IS THE SAME LOGICAL INCONSISTENCY AS ABOVE !!!!
C
         POINT  = FQFRST
         CALL REMOVE(FQ,POINT)
         PRT(POINT) = SRVBLK
         RMS(POINT) = REMSTG
         CALL FILE(SQ,POINT,4)
 4250    IF ( SQNUM .LE. 0 ) GOTO 9999
C
         POINT = SQFRST
 4300    K = TYPE(PRT(POINT))
         IF ( MU(K,PHASE) .GT. 0.0 ) GOTO 4400
 4350    POINT = SUCC(POINT)
         IF ( POINT .NE. 0 ) GOTO 4300
         REMSTG = 0
         GOTO 9999
C
C-----   EQUIPMENT IS REPAIRABLE
C
 4400    CALL REMOVE(SQ,POINT)
         CALL FILE (FQ,POINT,1)
C
         IF ( SPARES(K) .GT. 0 ) GOTO 4450
C
C-----   SPARES NO LONGER AVAILABLE; INCREMENT SPARE OUTAGE COUNTER
C
         NOSPRE(K) = NOSPRE(K) + 1
         GOTO 4350
C
C-----   SPARES STILL AVAILABLE
C
 4450    SRVMU  = MU(K,PHASE)
         RATE   = RATE  +  SRVMU * NSTAG
         REMSTG = RMS(POINT)
         SRVBLK = PRT(POINT)
         RTIME  = TIME
C
 9999    RETURN
         END
C
```

```
C ***********************************************************************
C ***********************************************************************
C ***********************************************************************
C
      SUBROUTINE POWERCHK
C                                                                 5/2/92
C-----------------------------------------------------------------------
C    PURPOSE:   TO BUILD AN ARRAY OF LOAD TO SOURCE POWER PATHS FOR
C               ALL POSSIBLE PATHS IN THE SYSTEM.
C    CALLED IN: MAIN
C    CALLS TO:
C-----------------------------------------------------------------------
C
C-----   FILE ARRAYDIM.INC CONTAINS THE MAXIMUM ARRAY DIMENSIONS AND
C        FILE DECLARE.INC  CONTAINS GLOBAL DECLARATIONS
C
        INCLUDE ARRAYDIM.INC
        INCLUDE DECLARE.INC
C
C -----     CALL CHKLOAD TO DETERMINE WHICH LOADS ARE NOT POWERED,
C -----     WHICH GENERATORS ARE PARALLELLED, AND GENERATOR LOADING.
C
      CALL CHKLOAD
C
C -----     CHECK THE LOADS WHICH WERE NOT POWERED. IF THE LOADING IS
C -----     CHANGED, THEN GO BACK TO THE BEGINNING OF THE SUBROUTINE TO
C -----     REDO THE LOADING CHECK.  MAKE SURE THAT ANY COMPONENT
C -----     RECONFIGURATION IS PROPERLY ACCOUNTED FOR USING THE STAT
C -----     ROUTINES, ETC.  THE LOADS ARE CHECKED BEFORE THE SOURCES
C -----     TO PREVENT LOAD SHEDDING OF HIGH PRIORITY LOADS.
C
      DO 4000 J = 1, NUMLD
C
C -----     IF THE LOAD NUMBERED J IS OK, IT'S ENTRY IN LOADWN IS 0.
C
        IF ( LOADWN(J) .EQ. 0 ) GOTO 4000
C
        WRITE(OUTPUT, 4100) LOADWN(J)
 4100   FORMAT(/'LOAD ',I4,' NOT POWERED, BUT IT SHOULD BE.')
 4000 CONTINUE
C
      DO 5000 I = 1, NUMSRC
C
C -----     IF THE GENERATOR IS NOT OVERLOADED, SKIP OVER IT
C
        IF ( OVRLD(I) .EQ. 0.0 ) GOTO 5000
        WRITE(OUTPUT, 5100) I
 5100   FORMAT(/'SOURCE(',I4,') IS OVERLOADED AND NEEDS ATTENTION.')
 5000 CONTINUE
C
      RETURN
C
      END
C
```

```
C ***********************************************************************
C ***********************************************************************
C ***********************************************************************
C
         SUBROUTINE POWERPATH
C                                                               5/6/92
C----------------------------------------------------------------------
C    PURPOSE:   TO BUILD AN ARRAY OF LOAD TO SOURCE POWER PATHS FOR
C               ALL POSSIBLE PATHS IN THE SYSTEM.
C    CALLED IN: MAIN
C    CALLS TO:
C----------------------------------------------------------------------
C
C-----   FILE ARRAYDIM.INC CONTAINS THE MAXIMUM ARRAY DIMENSIONS AND
C        FILE DECLARE.INC  CONTAINS GLOBAL DECLARATIONS
C
         INCLUDE ARRAYDIM.INC
         INCLUDE DECLARE.INC
C
C-----   LOCAL DECLARATIONS
C
      INTEGER      SP, TEMPATH, PATH, FLAGI, FLAGBLK
C
      DIMENSION  TEMPATH(LEN17*LEN23, LEN08), FLAGI(LEN15),
     1                FLAGBLK(LEN15), LCHK(LEN17*LEN23),
     2           TSRCPATH(((LEN18-LEN23)+2*LEN23)*LEN17, LEN08)
C
      NROWCT = 0
C
      DO 2700 I = 1, ((NUMLD-NUMABT)+2*NUMABT)*NUMSRC
        DO 2710 J = 1, NPARTS
          TSRCPATH(I,J) = 0
 2710   CONTINUE
 2700 CONTINUE
C
         DO 1000 J=1,NUMLD
C
C-----   INITIALIZE THE STACK POINTER
C
         SP = 0
         K = 0
         I = 0
         PATH = 1
C
      DO 1005 NN = 1, NPARTS
        DO 1015 N = 1, NUMSRC*NUMABT
          TEMPATH(N,NN) = 0
 1015   CONTINUE
 1005 CONTINUE
C
      DO 1025 N = 1, NUMSRC*NUMABT
        LCHK(N) = 0
 1025 CONTINUE
C
        DO 1035 N = 1, LEN15
          FLAGBLK(N) = 0
          FLAGI(N) = 0
 1035   CONTINUE
```

```
C
         BLK = LOAD(J)
C
C     WRITE(CONSOL, 90910) BLK
C     WRITE(OUTPUT, 90910) BLK
C90910      FORMAT('BUILDING STRINGS FOR LOAD ',I4)
C
      IND = FSTAR(BLK)
      BLK = ARC(IND)
C
C
 1010    CONTINUE
         IBEG = RSTAR(BLK)
         IEND = RSTAR(BLK+1) - 1
         IND  = IBEG
C
C-----   BLOCK IS AN EQUIPMENT, SO GO TO 1060
C
         IF ( IBEG .LE. IEND ) GOTO 1030
C
         GOTO 1060
C
C-----   BLOCK IS A GROUP
C
C -----     CHECK FOR COMPLETION OF HORIZONTAL MOVE
C
 1030    IF ( IND .GT. IEND ) GOTO 1060
C
         TBLK = RARC(IND)
C
      IF (TBLK .LE. LEN08) GOTO 1080
C
C
      NEEDBEG = RSTAR(TBLK)
      NEEDEND = RSTAR(TBLK+1)
      IF (NONEED(TBLK-LOWGRP+1) .LT. (NEEDEND-NEEDBEG)) THEN
        K = K+1
        FLAGBLK( K ) = TBLK
        FLAGI( K ) = I
      ELSE
      ENDIF
C
 1080    SP = SP + 4
         IF ( SP .GT. LEN12 ) THEN
           WRITE (OUTPUT,1040)
           WRITE (CONSOL,1040)
 1040      FORMAT(/' ERROR FOUND IN  SUBROUTINE POWERPATH')
           CALL ERRKDE(10,SP,LEN12)
         ENDIF
         STAK(SP-3) = BLK
         STAK(SP-2) = IBEG
         STAK(SP-1) = IEND
         STAK(SP)   = IND
         BLK = TBLK
         GOTO  1010
C
C-----   UPWARD RETRACE
C
```

```
 1060    IF ( SP .EQ. 0 ) GOTO 1099
C
      IF (BLK .GT. LEN08) GOTO 1072
C
C
      I = I+1
      TEMPATH( PATH, I ) = BLK
      DO 1071 M = 1, NUMSRC
        IF (BLK .EQ. SOURCE(M)) THEN
          PATH = PATH + 1
        ELSE
        ENDIF
 1071 CONTINUE
C
 1072    TBLK = BLK
         IND  = STAK(SP)
         IEND = STAK(SP-1)
         IBEG = STAK(SP-2)
         BLK  = STAK(SP-3)
         SP   = SP - 4
C
      IF (BLK .EQ. FLAGBLK( K )) THEN
        I = FLAGI( K )
        K = K-1
      ELSE
      ENDIF
C
C
C-----   INCREMENT INDEX OF ARC'S TAIL
C
 1090    IND = IND + 1
         GOTO 1030
C
 1099    CONTINUE
C
C
      NROWS = 0
      DO 2000 L = 1, NUMSRC*NUMABT
        LCHK( L ) = 0
          DO 2001 LL = 1, NPARTS
            LCHK( L ) = LCHK( L ) + TEMPATH( L, LL)
 2001     CONTINUE
        IF (LCHK( L ) .NE. 0) NROWS = NROWS + 1
 2000 CONTINUE
C
C
      DO 2500 L = 2, NUMSRC*NUMABT
        LL = 1
 2200   IF (TEMPATH( L, LL ) .NE. 0) GOTO 2500
        TEMPATH( L, LL) = TEMPATH( L-1, LL )
        LL = LL+1
        IF (LL .GT. NPARTS) GOTO 2500
        GOTO 2200
 2500 CONTINUE
C
      DO 3000 L = 1, NROWS
        DO 3010 LL = 1, NPARTS
        PWRPATH(NROWCT+L, LL) = TEMPATH(L, LL)
```

```
 3010   CONTINUE
 3000 CONTINUE
C
      NROWCT = NROWCT+NROWS
C
C
 1000 CONTINUE
C
      NNN = ((NUMLD-NUMABT) + 2*NUMABT)*NUMSRC
C
C -----     CREATE AN IMAGE OF PWRPATH FOR USE IN A SEARCH FOR CONTINUITY
C -----     FROM SOURCE TO LOAD FOR LOAD SHED INVESTIGATION.
C
      DO 9000 I = 1, NNN
        DO 9010 J = 1, NPARTS
          IF (PWRPATH(I, NPARTS-J+1) .NE. 0) THEN
            TSRCPATH(I,J) = PWRPATH(I, NPARTS-J+1)
          ELSE
            GOTO 9010
          ENDIF
 9010   CONTINUE
 9000 CONTINUE
C
      DO 9050 I = 1, NNN
        K = 1
        DO 9060 J = 1, NPARTS
          IF (TSRCPATH(I,J) .NE. 0) THEN
            SRCPATH(I,K) = TSRCPATH(I,J)
            K = K + 1
          ELSE
            GOTO 9060
          ENDIF
 9060   CONTINUE
 9050 CONTINUE
C
      WRITE(OUTPUT,5950)
 5950 FORMAT(/'TRANSPOSE OF POWER PATH CONTINUITY ARRAY'/)
C
      DO 5000 J = 1, NPARTS
      NOTHING = 0
        DO 5010 N = 1, NNN
          NOTHING = NOTHING + PWRPATH(N,J)
 5010   CONTINUE
      IF (NOTHING .NE. 0) THEN
         WRITE(CONSOL,5900) (PWRPATH(I,J),I=1,NNN)
         WRITE(OUTPUT,5900) (PWRPATH(I,J),I=1,NNN)
 5900       FORMAT(24I3)
      ELSE
         GOTO 5555
      ENDIF
C
 5000 CONTINUE
C
 5555 WRITE(OUTPUT,8950)
 8950 FORMAT(/'TRANSPOSE OF SOURCE PATH CONTINUITY ARRAY'/)
C
      DO 8500 J = 1, NPARTS
      NOTHING = 0
```

```
        DO 8510 N = 1, NNN
          NOTHING = NOTHING + SRCPATH(N,J)
 8510   CONTINUE
      IF (NOTHING .NE. 0) THEN
         WRITE(CONSOL,8900) (SRCPATH(I,J),I=1,NNN)
         WRITE(OUTPUT,8900) (SRCPATH(I,J),I=1,NNN)
 8900       FORMAT(24I3)
      ELSE
         GOTO 8000
      ENDIF
C
 8500 CONTINUE
C
 8000 RETURN
C
      END
C
C **********************************************************************
C **********************************************************************
C **********************************************************************
C
         FUNCTION RAND (SEED)
C                                                              02/25/88
C-----------------------------------------------------------------------
C     PURPOSE:   GENERATES A RANDOM NUMBER
C     CALLED IN: GENVNT
C-----------------------------------------------------------------------
C
         INTEGER A, B15, B16, FHI, P, SEED, XALO, XHI
C
         DATA A/16807/,B15/32768/,B16/65536/,P/2147483647/
C
C-----------------------------------------------------------------------
C
         XHI = SEED / B16
         XALO = ( SEED - XHI * B16 ) * A
         LEFTLO = XALO / B16
         FHI = XHI * A + LEFTLO
         K = FHI / B15
         SEED = (( (XALO - LEFTLO*B16) - P) + (FHI-K*B15) * B16 ) + K
         IF ( SEED .LT. 0) SEED = SEED + P
         RAND  =  FLOAT(SEED) * 4.656612875E-10
         RETURN
         END
C
```

```
C ***********************************************************************
C ***********************************************************************
C ***********************************************************************
C
          SUBROUTINE RCABLE (I)
C                                                               4/15/92
C-----------------------------------------------------------------------
C    PURPOSE:   FINDS THE CLOSEST COORDINATE FROM THE HIT TO THE CABLE.
C              SINCE THE CABLE IS CONSIDERED A LINE, THE FORMULA OF THE
C              DISTANCE FROM A POINT TO A LINE IS USED WITH A CHECK
C              TO SEE IF IT IS PERPENDICULAR.  THE CASE WHERE THE
C              LINE SEGMENT IS LOCATED SUCH THAT AN END POINT IS
C              CLOSEST IS CONSIDERED AND PROPERLY ACCOUNTED FOR.
C
C    CALLED IN: MAIN
C    CALLS TO:
C-----------------------------------------------------------------------
C
C-----    FILE ARRAYDIM.INC CONTAINS THE MAXIMUM ARRAY DIMENSIONS AND
C         FILE DECLARE.INC  CONTAINS GLOBAL DECLARATIONS
C
          INCLUDE ARRAYDIM.INC
          INCLUDE DECLARE.INC
C
C-----    LOCAL DECLARATIONS
C
      AA = CX2 - CX1
      BB = CY2 - CY1
      CC = CZ2 - CZ1
C
         TTT = AA*(HITX-CX1)+BB*(HITY-CY1)+CC*(HITZ-CZ1)
      TT = TTT/(AA**2+BB**2+CC**2)
C
            XX = AA * TT + CX1
      YY = BB * TT + CY1
      ZZ = CC * TT + CZ1
C
13050 IF ( XX .GE. CX1 ) .AND. ( XX .LE. CX2 ) GOTO 13100
      GOTO 13300
13100 IF ( YY .GE. CY1 ) .AND. ( YY .LE. CY2 ) GOTO 13200
      GOTO 13300
13200 IF ( ZZ .GE. CZ1 ) .AND. ( ZZ .LE. CZ2 ) GOTO 13400
13300 CXX = CX1
      CYY = CY1
      CZZ = CZ1
      RADSQ = (HITX-CXX)**2+(HITY-CYY)**2+
     &            (HITZ-CZZ)**2
      RAD1 = SQRT(RADSQ)
      CXX = CX2
      CYY = CY2
      CZZ = CZ2
      RADSQ = (HITX-CXX)**2+(HITY-CYY)**2+
     &            (HITZ-CZZ)**2
      RAD2 = SQRT(RADSQ)
      IF ( RAD2 .GT. RAD1 ) THEN
          RADIUS ( I ) = RAD1
      ELSE
          RADIUS ( I ) = RAD2
```

```
      ENDIF
      GOTO 13500
C
13400 CXX = XX
      CYY = YY
      CZZ = ZZ
      RADSQ = (HITX-CXX)**2+(HITY-CYY)**2+
     &              (HITZ-CZZ)**2
      RAD = SQRT(RADSQ)
      RADIUS ( I ) = RAD
C
C
13500         RETURN
C
      END
C
```

```
C ************************************************************************
C ************************************************************************
C ************************************************************************
C
        SUBROUTINE READIT
C                                                                 5/6/92
C-----------------------------------------------------------------------
C    PURPOSE:   READS IN USER SUPPLIED INFORMATION.
C    CALLED IN: MAIN
C    CALLS TO:  ERRDKE, SSORT
C    NOTE:      INPUT FORMAT IS SIMILAR TO TIGER'S.
C-----------------------------------------------------------------------
C
C-----  FILE ARRAYDIM.INC CONTAINS THE MAXIMUM ARRAY DIMENSIONS AND
C       FILE DECLARE.INC  CONTAINS GLOBAL DECLARATIONS
C
        INCLUDE ARRAYDIM.INC
        INCLUDE DECLARE.INC
C
C-----  LOCAL DECLARATIONS
C
        INTEGER  BUFF(10),    GN,    NM(4)
      INTEGER      JQ, KQ, J, I, SP
C
        DATA IBLANK/'    '/
C
C
C-----  END OF LOCAL DECLARATIONS
C
C-----------------------------------------------------------------------
C
C-----  INITIALIZE COUNTERS
C
        MAXLST = LEN11
        NARC   = 0
        NBLKS  = 0
        NEQT   = 0
        NGRCHG = 0
        NPARTS = 0
        NVDC   = 0
C
C -----     READ IN THE CODE SWITCH FOR SURVIVABILITY ANALYSIS
C -----     SURVIVE = 1  PERFORM SURVIVABILITY ANALYSIS
C -----     SURVIVE = 0  DO NOT PERFORM SURVIVABILITY ANALYSIS
C
      READ ( INPUT,91002 ) SURVIVE
      WRITE( OUTPUT, 91002) SURVIVE
91002 FORMAT(I2)
C
C-----  INPUT SIMULATION PARAMETERS
C
        READ (INPUT,90000) NMS
        WRITE (OUTPUT,90010) NMS
C
        READ (INPUT,90020) NMISS,MISSPR,LOWGRP,NSDEV,NPT,
     &                     SEED1,SEED2,SEED3,SEED4
C
        WRITE (OUTPUT,90040) NMISS,MISSPR,LOWGRP,NSDEV,NPT,
```

```
     &                         SEED1,SEED2,SEED3,SEED4
C
C
C
C-----   READ IN MISSION PROFILE ( TIME LINE )
C
         K = 0
C
 1000    READ (INPUT,90050) ( PHZ(K+J),PHTIME(K+J), J = 1,5 )
C           WRITE (OUTPUT,90050) ( PHZ(K+J),PHTIME(K+J),J = 1,5 )
         K = K + 5
         IF ( PHZ(K-4).NE.0 ) GOTO 1000
         NPHASE = K - 5
C
C-----   ACCUMULATE PHASE DURATIONS TO CONVERT INTO PHASE ENDING TIMES
C
         WRITE (OUTPUT,90070)
         DURMIS = 0.
         K = 1
C
 1010    IF ( PHZ(K) .EQ. 0 ) GOTO 1020
            DURMIS = DURMIS + PHTIME(K)
            WRITE (OUTPUT,90080) K,PHZ(K),PHTIME(K),DURMIS
            PHTIME(K) = DURMIS
            K = K + 1
         GOTO 1010
C
C-----   INPUT REPORT OPTIONS
C
 1020    READ (INPUT,90090) KOPT
         WRITE (OUTPUT,90100) KOPT
C
         IF ( KOPT.GE.4 ) WRITE (OUTPUT,90110)
C
C-----   INPUT SERVER STATUS FOR DIFFERENT PHASE TYPES
C
         READ (INPUT,90090) ( PHSRV(I), I=1,NPT )
         IF ( KOPT.LT.4 ) GOTO 1100
C
         DO 1050 I = 1,NPT
            WRITE (OUTPUT,90120) I,PHSRV(I)
 1050    CONTINUE
C
 1100    READ (INPUT,90130) NSTAG
C
         IF ( NSTAG.LE.0 ) NSTAG = 1
         IF ( KOPT.GE.4 ) WRITE (OUTPUT,90140) NSTAG
C
C-----   START EQUIPMENT PARAMETER INPUT BY TYPE
C
         IF ( KOPT.GE.4 ) WRITE (OUTPUT,90150)
C
 2000    READ (INPUT,90160) I,NM,X,Y,U,IPRI
            IF ( I.EQ.0 )   GOTO 2999
            IF ( KOPT.GE.4 )
     1            WRITE (OUTPUT,90170) I,NM,X,Y,U,IPRI
            PRI(I) = IPRI
            IF ( U.EQ.0.0 ) U = 1.0
```

```
          IF ( U.GE.0.0 ) GOTO 2050
          U = 1.0
C
C-----    READ VARIABLE DUTY CYCLE INFO IF U<0
C
          NVDC = NVDC + 1
          IF ( NVDC.GT.LEN04 ) THEN
             WRITE (OUTPUT,2025)
             WRITE (CONSOL,2025)
 2025        FORMAT(/' ERROR FOUND IN SUBROUTINE READIT')
             CALL ERRKDE(1,NVDC,LEN04)
          ENDIF
          IVDC(I) = NVDC
          READ (INPUT,90180) ( VDC(NVDC,K), K = 1,NPT )
          IF ( KOPT.GE.4 ) WRITE (OUTPUT,90190) I,NM,
     &                          ( VDC(NVDC,K), K =1,NPT )
C
 2050     CONTINUE
C
          IF ( I.GT.LEN07 ) THEN
             WRITE (OUTPUT,2025)
             WRITE (CONSOL,2025)
             CALL ERRKDE(2,I,LEN07)
          ENDIF
          NEQT = MAX0(NEQT,I)
C
          DO 2080 J = 1,4
             NAME(I,J) = NM(J)
 2080     CONTINUE
C
          LAM(I) = 0.0
          IF ( X.NE.0.0 ) LAM(I) = 1.0 / X * U
          IF ( Y.GE.0.0 ) GOTO 2200
C
C-----    READ VARIABLE MTTR CARD
C
          READ (INPUT,90200) ( MU(I,J), J = 1,NPT )
          IF ( KOPT.GE.4 )
     1       WRITE (OUTPUT,90210) I,NM, ( MU(I,J), J = 1,NPT )
C
C
          DO 2150 J = 1,NPT
             IF ( MU(I,J).LE.0.0 .OR. MU(I,J).GE.9998.0 ) GOTO 2120
             MU(I,J) = 1.0 / MU(I,J)
             GOTO 2150
 2120        MU(I,J) = 0.0
 2150     CONTINUE
          GOTO 2000
C
 2200     DO 2250 J = 1,NPT
             IF ( Y.EQ.0.0 .OR. Y.GE.9999.0 ) GOTO 2220
             MU(I,J) = 1.0 / Y
             GOTO 2250
 2220        MU(I,J) = 0.0
 2250     CONTINUE
       GOTO 2000
C
C-----  END EQUIPMENT PARAMETER INPUT
```

```
C
 2999    CONTINUE
C
C-----   START EQUIPMENT TYPE-TO-NUMBER MAP INPUT
C
         IF ( KOPT.GE.4 ) WRITE (OUTPUT,90220) NMS
 4000    READ (INPUT,90230) I,BUFF
            IF ( I.EQ.0 )    GOTO 4900
            IF ( I.GT.LEN07 ) THEN
               WRITE (OUTPUT,2025)
               WRITE (CONSOL,2025)
               CALL ERRKDE (3,I,LEN07)
            ENDIF
            IF ( KOPT.GE.4) WRITE (OUTPUT,90240) I,
     &                         ( NAME(I,J), J = 1,4 ),BUFF
            J = 1
 4200       IF ( BUFF(J).EQ.0 ) GOTO 4000
            IF ( BUFF(J).GT.LEN08 ) CALL ERRKDE (4,BUFF(J),LEN08)
            NPARTS = MAX0(NPARTS,BUFF(J))
            TYPE(BUFF(J)) = I
            J = J + 1
            IF ( J.LE.10 ) GOTO 4200
         GOTO 4000
C
C-----   END EQUIPMENT TYPE-TO-NUMBER MAP INPUT
C
 4900    CONTINUE
         LOWGRP = MAX0(LOWGRP,NPARTS + 1)
C
         NP1 = NPARTS + 1
         NP2 = NPARTS + 2
C
C-----   READ IN SPARES POLICY
C
         READ (INPUT,90250) IUNLIM
         IF ( IUNLIM .NE. IBLANK ) GOTO 5100
C
         IF ( KOPT .GE. 4 ) WRITE (OUTPUT,90260)
C
C -----     NEED TO HAVE NO SPARES FOR THE SURVIVABILITY ANALYSIS
C -----     SO MAKE ALL SPARES AVAILABLE EQUAL TO ZERO.
C
         DO 5000 I = 1, NEQT
            READ (INPUT,90270) L,K
            IF ( K .EQ. 0 ) K = I
            SPARE1(K) = L
            IF ( KOPT .GE. 4 )
     &         WRITE (OUTPUT,90280) K,( NAME(K,J), J = 1,4 ),SPARE1(K)
 5000    CONTINUE
         GOTO 5200
C
C-----   UNLIMITED SPARES
C
 5100    CONTINUE
         DO 5150 I = 1, NEQT
            SPARE1(I) = 999999
 5150    CONTINUE
C
```

```
      IF ( KOPT .GE. 4 ) WRITE (OUTPUT,90290)
C
 5200 CONTINUE
C
C-----INPUT SYSTEM CONFIGURATION
C
      IF ( KOPT.GE.4 ) WRITE (OUTPUT,90300)
C
      READ (INPUT,90310) ISYSNM,ISYS
      IF ( KOPT.GE.4 ) WRITE (OUTPUT,90320) ISYSNM,ISYS
C
      IF ( KOPT.GE.4 ) WRITE (OUTPUT,90350)
C
C-----START GROUP CONFIGURATION INPUT
C
 7000 READ (INPUT,90360) NR,GN,( BUFF(I),I = 1,10 )
         IF ( KOPT.GE.4 .AND. GN.NE.0 )
     &       WRITE (OUTPUT,90370) GN,NR,( BUFF(I), I = 1, 10 )
         IF ( NR.EQ.0 ) GOTO 7999
C
C-----   INPUT VARIABLE GROUP REQUIREMENTS BY PHASE ( IF NR < 0 )
C
         IF ( NR. GT. 0  ) GOTO 7050
C
         NGRCHG = NGRCHG + 1
         GRPCHG(NGRCHG) = GN
         READ (INPUT,90090) ( GRPREQ(NGRCHG,I), I = 1, NPT )
         IF ( KOPT .GE. 4 )
     1      WRITE (OUTPUT,90380) GN,( GRPREQ(NGRCHG,I), I = 1, NPT )
         NR = 1
C
 7050    IF ( GN .GT. LEN15  ) THEN
            WRITE (OUTPUT,2025)
            WRITE (CONSOL,2025)
            CALL ERRKDE( 5,GN,LEN15)
         ENDIF
         IF ( GN .LT. LOWGRP ) THEN
            WRITE (OUTPUT,2025)
            WRITE (CONSOL,2025)
            CALL ERRKDE(13,GN,LOWGRP)
         ENDIF
         GCOUNT = GN - LOWGRP + 1
         NONEED(GCOUNT) = NR
         NBLKS  = MAX0(NBLKS,GN)
C
C-----   ESTABLISH THE HIERARCHICAL RELIABILITY FAULT TREE OF THE
C        SYSTEM BY FORMING THE FUNCTIONS ARC, RARC, FSTAR, AND
C        RSTAR.  ARC AND FSTAR MOVE FROM THE BOTTOM UP;
C        RARC AND RSTAR MOVE FROM THE TOP DOWN.
C
         DO 7100 I = 1,10
            IF ( BUFF(I) .EQ. 0 ) GOTO 7000
            NARC = NARC + 1
            IF ( NARC.GT.LEN10 ) CALL ERRKDE(6,NARC,LEN10)
            ARC(NARC)  = GN
            RARC(NARC) = BUFF(I)
            RSTAR(GN)        = RSTAR(GN)        + 1
            FSTAR(BUFF(I)) = FSTAR(BUFF(I)) + 1
```

```
                NOOFF(GCOUNT)   = NOOFF(GCOUNT)   + 1
C
C-----          N1 AND N2 TEMPORARILY HOLD THE TAILS AND HEADS
C                  OF THE GROUP-MEMBER ARCS PRIOR TO SORTING
C
                N1(NARC) = BUFF(I)
                N2(NARC) = GN
 7100        CONTINUE
          GOTO 7000
C
C
C-----    END GROUP CONFIGURATION INPUT
C
 7999     CONTINUE
          NGROUP = NBLKS - LOWGRP + 1
C
C
C-----    END GROUP CONFIGURATION INPUT
C
C -----      ADD SWITCH TO JUMP OVER HANDLER WHICH READS IN SURVIVABILITY
C -----      DATA WHEN RELIABILITY-ONLY RUNS ARE DESIRED.
C
 9700 IF ( SURVIVE .EQ. 0 ) GOTO 9790
C
C -----      INITIALIZE SOME COUNTERS
C
      KQ = 10
      KC = 10
      EQUIPNO = 0
      EQCODE =  0
      NUMLD =   0
      NUMSRC =  0
      NUMCBL =  0
      NUMCAB =  0
      NUMABT =  0
      NUMBKR =  0
      NUMBT =   0
      NUMALT =  0
      COSTEQ =  0
      COSTTL  = 0
C
C -----      READ IN THE DATA FOR THE COMPONENT LOCATION
C -----      AND COMPONENT DATA.  EQDATA() HOLDS THIS INFO.
C
 9705 COSTTL = COSTTL + COSTEQ
      READ ( INPUT, 90400 ) EQCODE, EQUIPNO,
     &                ( EQTEMP(JQ), JQ = 1,KQ )
90400 FORMAT (2I4, 10F8.2 )
C
C -----      INDICATES THE END OF THE INPUT DATA
C
      IF ( EQCODE .EQ. 9999 ) GOTO 9790
C
      IF ( EQCODE .LE. 0 ) GOTO 9705
C
C -----      KEEP AN ARRAY OF EQUIPMENT TYPES
C
      EQTYPE ( EQUIPNO ) = EQCODE
```

```
C
      GOTO (9710,9720,9730,9740,9750,9760,9770,9780), EQCODE
C
C
C -----     CODING OF THE EQuipment CODE:
C
C     EQCODE       = 1   ALTERNATE DC POWER SOURCE DATA
C     EQCODE       = 2   POWER SOURCE DATA
C     EQCODE       = 3   LOAD DATA
C     EQCODE       = 4   BREAKER DATA
C     EQCODE       = 5   CABLE DATA
C     EQCODE       = 6   CABINET DATA
C     EQCODE       = 7   BUS TIE DATA
C     EQCODE       = 8   ABT DATA
C
C
C
C -----     EQCODE = 1 IS FOR ALTERNATE OR EMERGENCY POWER SOURCES SUCH
C -----     AS UPS, BATTERIES, OR FUEL CELLS.
C
 9710 NUMALT=NUMALT+1
      ALTSRC(NUMALT) = EQUIPNO
      DO 9715 JQ = 1,KQ
      EQDATA(EQUIPNO, JQ) = EQTEMP(JQ)
 9715 CONTINUE
      COSTEQ = EQTEMP(KC)
      GOTO 9705
C
C -----     EQCODE OF 2 INDICATES A SOURCE DATA FIELD.  THE FIRST THREE
C -----     NUMBERS ARE THE X, Y, AND Z LOCATION OF THE SOURCE.  THE NEXT
C -----     NUMBER INDICATES THE MAX POWER AVAILABLE.  THE NEXT FIELD IS
C -----     THE INITIAL STATUS OF THE SOURCE.  THE LAST FIELD INDICATES
C -----     THE CURRENT STATUS OF THE SOURCE.
C -----     THE SOURCE() ARRAY KEEPS TRACK OF SOURCE RATINGINFORMATION.
C
 9720 NUMSRC=NUMSRC+1
      SOURCE(NUMSRC) = EQUIPNO
      DO 9725 JQ = 1,KQ
      EQDATA(EQUIPNO, JQ) = EQTEMP(JQ)
 9725 CONTINUE
      COSTEQ = EQTEMP(KC)
      GOTO 9705
C
C -----     EQCODE OF 3 INDICATES A LOAD DATA FIELD.  THE FIRST THREE
C -----     NUMBERS ARE THE X, Y, AND Z LOCATION OF THE LOAD.  THE NEXT
C -----     NUMBER INDICATES THE POWER REQUIRED. THE LOAD() ARRAY
C -----     KEEPS TRACK OF THE LOAD RATING INFORMATION.
C
 9730 NUMLD=NUMLD+1
      LOAD(NUMLD) = EQUIPNO
      DO 9735 JQ = 1,KQ
      EQDATA(EQUIPNO, JQ) = EQTEMP(JQ)
 9735 CONTINUE
      COSTEQ = EQTEMP(KC)
      GOTO 9705
C
C -----     EQCODE OF 4 INDICATES A BREAKER DATA FIELD.  THE FIRST THREE
C -----     NUMBERS ARE THE X, Y, AND Z LOCATION OF THE BREAKER.  THE NEXT
```

```
C -----     NUMBER INDICATES THE PRIORITY LEVEL FOR SELECTIVE TRIPPING
C -----  DECISIONS.  THE LAST TWO FIELDS INDICATE THE INITIAL AND
C -----     CURRENT STATUS OF THE BREAKER RESPECTIVELY.
C -----     THE BREAKER() ARRAY KEEPS TRACK OF THE BREAKER INFORMATION.
C
 9740 NUMBKR=NUMBKR+1
      BREAKER(NUMBKR) = EQUIPNO
      DO 9745 JQ = 1,KQ
      EQDATA(EQUIPNO, JQ) = EQTEMP(JQ)
 9745 CONTINUE
      COSTEQ = EQTEMP(KC)
      GOTO 9705
C
C -----     EQCODE OF 5 INDICATES A CABLE DATA FIELD.  THE FIRST THREE
C -----     NUMBERS ARE THE X, Y, AND Z LOCATION OF ONE END OF THE CABLE.
C -----  THE NEXT THREE NUMBERS ARE THE COORDINATES OF THE OTHER END.
C -----     THE TWO ENDS INDICATE A LINE SEGMENT WHICH MUST HAVE DISTANCE
C -----     TO ANY POINT ALONG THE CABLE FOR DAMAGE CHECK.
C
 9750 NUMCBL=NUMCBL+1
      CABLE(NUMCBL) = EQUIPNO
      DO 9755 JQ = 1,KQ
      EQDATA(EQUIPNO, JQ) = EQTEMP(JQ)
 9755 CONTINUE
      CALL LCABLE(EQUIPNO)
      COSTEQ = CBLENGTH*EQTEMP(KC)
      GOTO 9705
C
C -----     EQCODE OF 6 INDICATES A CABINET DATA FIELD.  THE FIRST THREE
C -----     NUMBERS ARE THE X, Y, AND Z LOCATION OF THE CABINET.  THE NEXT
C -----     THREE NUMBERS ARE NOT USED. THE CABINET() ARRAY
C -----     KEEPS TRACK OF THE CABINET INFORMATION.
C
 9760 NUMCAB=NUMCAB+1
      CABINET(NUMCAB) = EQUIPNO
      DO 9765 JQ = 1,KQ
      EQDATA(EQUIPNO, JQ) = EQTEMP(JQ)
 9765 CONTINUE
      COSTEQ = EQTEMP(KC)
      GOTO 9705
C
C -----     EQCODE OF 7 INDICATES A BUS TIE DATA FIELD.  THE FIRST THREE
C -----     NUMBERS ARE THE X, Y, AND Z LOCATION OF THE BUS TIE.  THE NEXT
C -----     NUMBER IS NOT USED. THE NEXT FIELD INDICATES THE INITIAL STATUS
C -----     OF THE BUS TIE.  THE LAST FIELD INDICATES THE CURRENT STATUS
C -----     OF THE BUS TIE.  THE BUSTIE() ARRAY
C -----     KEEPS TRACK OF THE LOAD RATING INFORMATION.
C
 9770 NUMBT=NUMBT+1
      BUSTIE(NUMBT) = EQUIPNO
      DO 9775 JQ = 1,KQ
      EQDATA(EQUIPNO, JQ) = EQTEMP(JQ)
 9775 CONTINUE
      COSTEQ = EQTEMP(KC)
      GOTO 9705
C
C -----     EQCODE OF 8 INDICATES AN ABT DATA FIELD.  THE FIRST THREE
C -----     NUMBERS ARE THE X, Y, AND Z LOCATION OF THE ABT.  THE NEXT
```

```
C -----     NUMBER INDICATES THE BUS ALIGNMENT OF THE ABT.
C -----  THE ABT() ARRAY KEEPS TRACK OF THE ABT
C -----  INFORMATION.
C
 9780 NUMABT=NUMABT+1
      ABT(NUMABT) = EQUIPNO
      DO 9785 JQ = 1,KQ
      EQDATA(EQUIPNO, JQ) = EQTEMP(JQ)
 9785 CONTINUE
      COSTEQ = EQTEMP(KC)
      GOTO 9705
C
 9790 CONTINUE
C
C -----     OUTPUTTING EQDATA ARRAY
C
      WRITE(OUTPUT, 9692)
 9692 FORMAT(/'EQDATA   I          DATA FIELDS 1 - 10'/)
         DO 9690 I = 1, NPARTS
          WRITE(OUTPUT, 9695) I,(EQDATA(I,J), J=1,10)
 9695     FORMAT('      ',I4,'   ',10F8.2)
 9690 CONTINUE
C
C -----     READ IN THE HIT DATA.  THE FIRST THREE NUMBERS
C -----     INDICATE THE X, Y, Z COORDINATES OF THE HIT.  THE NEXT TWO
C -----     NUMBERS ARE THE MIN AND MAX RADIUS OF THE HIT.
C
      READ( INPUT, 90500 ) ( HITDATA(JQ), JQ = 1,KQ )
90500 FORMAT (9F8.2 )
C
      NDISP = 0
12000 READ( INPUT, 90505 ) BUFF
90505 FORMAT(10I4)
      J = 1
12020 IF ((BUFF(J).EQ.0).AND.(NDISP .EQ. 0)) THEN
        DISPALL = 1
        GOTO 12900
      ELSE IF ((BUFF(J).EQ.0).AND.(J .EQ. 1)) THEN
        GOTO 12900
      ELSE IF (BUFF(J).EQ.0) THEN
        GOTO 12000
      ELSE
      ENDIF
C
      NDISP = NDISP + 1
      BLKDISP(NDISP) = BUFF(J)
C
      J = J + 1
      IF (J .LE. 10) GOTO 12020
C
      GOTO 12000
C
12900 CONTINUE
C
C
C-----    REORDER THE ARC LIST IN THE ASCENDING ORDER OF THE ORIGINAL
C            RARC LIST (I.E., N1)
C
```

```
      CALL SSORT( ARC,N1,NARC)
C
C-----  REORDER THE RARC LIST IN THE ASCENDING ORDER OF THE ORIGINAL
C          ARC LIST (I.E., N2)
C
      CALL SSORT(RARC,N2,NARC)
C
C-----  CLEAR THE USED PORTION OF THE N1 AND N2 ARRAYS BEFORE REUSE
C
      DO 9800 I = 1, NARC
         N1(I) = 0
         N2(I) = 0
 9800 CONTINUE
C
C-----  TRANSFORM FSTAR AND RSTAR BASED ON THE ARC COUNT
C
      ITF = FSTAR(1)
      ITR = RSTAR(1)
      FSTAR(1) = 1
      RSTAR(1) = 1
C
      DO 9900 I = 1,NBLKS
         ITF1 = FSTAR(I+1)
         ITR1 = RSTAR(I+1)
         FSTAR(I+1) = FSTAR(I) + ITF
         RSTAR(I+1) = RSTAR(I) + ITR
         ITF = ITF1
         ITR = ITR1
 9900 CONTINUE
C
C
      RETURN
C
90000 FORMAT(4X,19A4)
90010 FORMAT(/,1x,75('/'),//,10X,'RUN ID:',19A4)
90020 FORMAT(I5,I5,I5,F5.0,I4,4I12)
90040 FORMAT(/,5X,'NO. OF MISSIONS:',                T40,I12,
    1         /,5X,'MISSIONS BETWEEN SHORT PRINTS:',T40,I12,
    2         /,5X,'LOWEST GROUP NUMBER:',          T40,I12,
    3         /,5X,'NO. OF STD. DEVIATIONS:',       T40,F12.4,
    4         /,5X,'NO. OF PHASE TYPES:',           T40,I12,
    5         /,5X,'SEED NO. 1 :',                  T40,I12,
    6         /,5X,'SEED NO. 2 :',                  T40,I12,
    7         /,5X,'SEED NO. 3 :',                  T40,I12,
    8         /,5X,'SEED NO. 4 :',                  T40,I12)
90050 FORMAT(5(I2,F8.0))
C90060  FORMAT(1X,5(I2,F8.0))
90070 FORMAT(/,T10,'MISSION TIME LINE',/,T10,'PHASE NO.',
    &           T27,'TYPE',T37,'DURATION',T51,'CUM. TIME')
90080 FORMAT(T10,I5,T25,I5,T35,F10.2,T50,F10.2)
90090 FORMAT(20I4)
90100 FORMAT(/,10X,'REPORT OPTION SELECTED : ',I5,/)
90110 FORMAT(/,T10,'SERVER STATUS CODE: 0=OPER. 1=NOT OPER.',//,
    &           T35,'SERVER STATUS',T10,'PHASE TYPE')
90120 FORMAT(T15,I4,T35,I4)
90130 FORMAT(20X,I4)
90140 FORMAT(/,T10,'NO. STAGES OF SERVICE:',T40,I11)
90150 FORMAT(/,1x,75('/'),//,10X,'EQUIPMENT TYPES AND PARAMETERS',/,
```

```
     1          1X,'TYPE     NAME',10X,'MTTF      MTTR',6X,'DUTYCYCLE',
     2             '    REP. PRIORITY')
90160     FORMAT(I4,4A4,F10.0,2F4.0,I4)
90170     FORMAT(1X,I4,4A4,F10.0,F8.2,5X,F6.2,3X,I10)
90180     FORMAT(20F4.0)
90190     FORMAT(' EQUIP TYPE',I5,4A4,/,' DUT CYC:',(10(2X,F6.4,2X),/))
90200     FORMAT(20F4.0)
90210     FORMAT(' EQUIP TYPE',I5,4A4,/,
     &            ' VAR REP:',(10(F7.2),/,10(F7.2)))
90220     FORMAT(/,1x,75('/'),//,19A4//1X,'EQUIP-TYPE EQUIP-NUMBER MAP',
     &            /1X,'TYPE      NAME',14X,
     &            'EQUIPMENT NUMBERS OF THIS TYPE')
90230     FORMAT(11I4)
90240     FORMAT(1X,I4,1X,4A4,10I4)
90250     FORMAT(A4)
90260     FORMAT(/,1x,75('/'),//,T10,'SPARES PROVISIONING POLICY',/,
     &                T10,' NO. ',T20,' EQUIP NAME ',T39,'SPARES')
90270     FORMAT(2I4)
90280     FORMAT(T10,I5,T20,4A4,T40,I5)
90290     FORMAT(' UNLIMITED SPARES')
90300     FORMAT(/,1x,75('/'),//,T10,'CONFIGURATION OF SYSTEM',/)
90310     FORMAT(A4,4X,I4)
90320     FORMAT(T10,'SYSTEM NAME:    ',T31,A4,
     1         /T10,'SYSTEM GROUP NO:',T30,I5,/,T10,/)
90330     FORMAT(A4,8X,I4)
90350     FORMAT(/,1x,75('/'),//,T10,'GROUP CONFIGURATIONS',/)
90360     FORMAT(12I4)
90370     FORMAT(' GROUP: ',I5,' REQUIRES',I4,' OF MEMBERS: ',10I5)
90380     FORMAT('  GRP:',I5,' REQ: ',20I5,/,10X,20I5)
C
          END
C
```

```
C ***********************************************************************
C ***********************************************************************
C ***********************************************************************
C
      SUBROUTINE REMOVE (Q,REC)
C                                                               10/03/88
C-----------------------------------------------------------------------
C   PURPOSE:   REMOVES THE RECORD REC FROM THE LIST Q
C   CALLED IN: FAIL, PCHANG, REPAIR
C   CALLS TO:  ERRKDE
C-----------------------------------------------------------------------
C
C-----   FILE ARRAYDIM.INC CONTAINS THE ARRAY DIMENSIONS AND
C        FILE DECLARE.INC CONTAINS GLOBAL DECLARATIONS
C
         INCLUDE ARRAYDIM.INC
         INCLUDE DECLARE.INC
C
C-----   LOCAL DECLARATIONS
C
         INTEGER  REC, Q
         DIMENSION Q(3)
C
C-----   END OF LOCAL DECLARATIONS
C
C-----------------------------------------------------------------------
C
C-----   DECREMENT LIST COUNT
C
         NINLST = NINLST - 1
         IF ( Q(3) .LE. 0 ) CALL ERRKDE (15,0,0)
         IF ( REC. NE. Q(1) ) GOTO 1200
         Q(1) = SUCC(Q(1))
         IF ( Q(1) .NE. 0 ) GOTO 1100
         Q(2) = 0
         GOTO 5000
C
 1100    PRED(Q(1)) = 0
         GOTO 5000
C
 1200    IF ( REC .NE. Q(2) ) GOTO 1400
         Q(2) = PRED(Q(2))
         IF ( Q(2) .NE. 0 ) GOTO 1300
         Q(1) = 0
         GOTO 5000
C
 1300    SUCC(Q(2)) = 0
         GOTO 5000
C
 1400    SUCC(PRED(REC)) = SUCC(REC)
         PRED(SUCC(REC)) = PRED(REC)
 5000    Q(3) = Q(3) - 1
C
         RETURN
         END
C
```

```
C ************************************************************************
C ************************************************************************
C ************************************************************************
C
      SUBROUTINE REPAIR (BLK)
C                                                                02/29/88
C-----------------------------------------------------------------------
C     PURPOSE:   CHECKS IF ANYTHING IS REPAIRABLE WITHIN THE PHASE
C     CALLED IN: MAIN
C     CALLS TO:  FILE, REMOVE
C     NOTE:      IN THE PRESENT SET-UP ONLY ONE STAGE OF REPAIR CAN BE
C                DONE/EVENT. THE MOST POPULAR CASE IN A REPAIR SCENARIO
C                IS A ONE-STAGE REPAIR IN WHICH CASE REMSTG IS EITHER 1
C                OR 0. OTHER CASES CAN BE CONSIDERED THOUGH.
C-----------------------------------------------------------------------
C
C-----   FILE ARRAYDIM.INC CONTAINS THE ARRAY DIMENSIONS AND
C        FILE DECLARE.INC CONTAINS GLOBAL DECLARATIONS
C
      INCLUDE ARRAYDIM.INC
      INCLUDE DECLARE.INC
C
C-----------------------------------------------------------------------
C
C-----   DECREMENT THE NUMBER OF STAGES OF SERVICE COMPLETED
C
      REMSTG = REMSTG - 1
      IF ( REMSTG .GT. 0 ) RETURN
C
C-----   END OF SERVICE NOW - COLLECT STATS
C
      SPARES(TYPE(BLK)) = SPARES(TYPE(BLK)) - 1
      NREP = NREP + 1
      RT   = TIME - RTIME
      RSUM = RSUM + RT
      RSQ  = RSQ + RT * RT
C
C-----   ADJUST SYSTEM EVENT RATE
C
      RATE   = RATE - SRVMU * NSTAG + LAM(TYPE(BLK))
C
C-----   RETURN IF SERVICE QUEUE IS EMPTY
C
      IF ( SQNUM .EQ. 0 ) RETURN
C
C-----   SEARCH QUEUE FOR A PART REPAIRABLE IN THIS PHASE;
C        RETURN IF NONE FOUND
C
      IPOINT = SQFRST
  100 K = TYPE(PRT(IPOINT))
      IF ( MU(K,PHASE) .GT. 0 ) GOTO 200
      IPOINT = SUCC(IPOINT)
      IF (IPOINT .NE. 0 ) GOTO 100
      RETURN
C
C-----   PART REPAIRABLE; REMOVE IT FROM QUEUE AND PLACE IN SERVICE
C
  200 CALL REMOVE(SQ,IPOINT)
```

```
      CALL FILE(FQ,IPOINT,1)
C
C-----  NOTE AGAIN THE SO-DESIGNATED POSITION OF IN-SERVICE REPAIR.
C
      IF ( SPARES(K).GT. 0 ) GOTO 300
C
C-----  SPARES ARE EXHAUSTED; INCREMENT NUMBER OF SPARE OUTAGES
C
      NOSPRE(K) = NOSPRE(K) + 1
      IPOINT = SUCC(IPOINT)
      IF (IPOINT .NE. 0 ) GOTO 100
      RETURN
C
C-----  SPARES STILL AVAILABLE; ADJUST SYSTEM EVENT RATE
C
  300 SRVMU  = MU(K,PHASE)
      RATE   = RATE + SRVMU * NSTAG
      REMSTG = RMS(IPOINT)
      SRVBLK = PRT(IPOINT)
      RTIME  = TIME
C
      RETURN
      END
C
```

```
C *********************************************************************
C *********************************************************************
C *********************************************************************
C
      SUBROUTINE REPORT
C                                                              03/16/88
C----------------------------------------------------------------------
C   PURPOSE:   GENERATES AND PRINTS FINAL REPORT
C   CALLED IN: MAIN
C   CALLS TO:  SORT
C----------------------------------------------------------------------
C
C-----  FILE ARRAYDIM.INC CONTAINS THE ARRAY DIMENSIONS AND
C       FILE DECLARE.INC CONTAINS GLOBAL DECLARATIONS
C
      INCLUDE ARRAYDIM.INC
      INCLUDE DECLARE.INC
C
C----------------------------------------------------------------------
C
C-------------------------
C-----  BASIC SYSTEM REPORT
C-------------------------
C
      RNMISS = 1.0 / FLOAT(NMISS)
      RDURMS = 1.0 / DURMIS
      IF ( N1(ISYS). LE. 0) GOTO 100
      RN1SYS = 1.0 / FLOAT(N1(ISYS))
C
 100  UTIME  = T1(ISYS) * RNMISS
      AV     = UTIME * RDURMS
      SDTIM  = DURMIS - UTIME
      RB     = 1.0 - FLOAT(N2(ISYS)) * RNMISS
      SMTTF  = -1.0
      IF ( NTFALS .GT. 0 ) SMTTF = DURMIS * NMISS / NTFALS
      SMTBF  = -1.0
      NSUC   = NMISS - N2(ISYS)
      IF ( N1(ISYS) .GT. 0) SMTBF  = T1(ISYS) * RN1SYS
C
      WRITE (OUTPUT,9010) NMS
      WRITE (OUTPUT,9000) NMISS,NSUC,RB,AV,UTIME,SDTIM,DURMIS,
     &                    N1(ISYS),SMTBF,SMTTF,NTFALS,NEVENT
C
C----------------------------
C-----  MAJOR SUBSYSTEM REPORT
C----------------------------
C
      WRITE (OUTPUT,9010) NMS
      WRITE (OUTPUT,9020)
C
      DO 125 I = 1,NSSYS
         UTIME = T1(ISSYS(I)) * RNMISS
         AV    = UTIME * RDURMS
         DTIME = DURMIS - UTIME
         RB    = 1.0 - FLOAT(N2(ISSYS(I))) * RNMISS
         X     = N1(ISSYS(I)) * RNMISS
         WRITE (OUTPUT,9030) ISSNAM(I),ISSYS(I),X,DTIME,RB,AV
 125  CONTINUE
```

```
C
        IF (KOPT .LT. 1) GOTO 1000
C
C------------------------------------------------
C-----  RELIABILITY REPORT FOR PARTS AND GROUPS
C------------------------------------------------
C
        WRITE (OUTPUT,9010) NMS
        WRITE (OUTPUT,9040)
C
        KWROT = 0
        DO 150 I = 1,NPARTS
           IF ( N1(I) .LE. 0) GOTO 150
           UTIME = T1(I) * RNMISS
           AV    = UTIME * RDURMS
           DTIME = DURMIS - UTIME
           RB    = 1.0 - FLOAT(N2(I)) * RNMISS
           X     = N1(I) * RNMISS
           WRITE (OUTPUT,9050) I,TYPE(I),( NAME(TYPE(I),J), J = 1,4),
     &                         X,DTIME,RB,AV
           KWROT = KWROT + 1
           IF ( MOD(KWROT,50) .NE. 0 ) GOTO 150
           IF ( I .GE. NPARTS ) GOTO 150
           WRITE (OUTPUT,9010) NMS
           WRITE (OUTPUT,9040)
  150   CONTINUE
C
        WRITE (OUTPUT,9010) NMS
        WRITE (OUTPUT,9060 )
C
        KWROT = 0
        DO 175 I = LOWGRP,NBLKS
           IF ( N1(I) .LE. 0 ) GOTO 175
           UTIME = T1(I) * RNMISS
           AV    = UTIME * RDURMS
           DTIME = DURMIS - UTIME
           RB    = 1.0 - FLOAT(N2(I)) * RNMISS
           X     = N1(I) * RNMISS
           WRITE (OUTPUT,9070) I,X,DTIME,RB,AV
           KWROT = KWROT + 1
           IF ( MOD(KWROT,50) .NE. 0 ) GOTO 175
           IF ( I .GE. NBLKS ) GOTO 175
           WRITE (OUTPUT,9010) NMS
           WRITE (OUTPUT,9060 )
  175   CONTINUE
C
C-------------------------------
C-----  CRITICAL EQUIPMENT REPORT
C-------------------------------
C
C-----  RELIABILITY HIT LIST BY EQUIPMENT NUMBER
C
 1000   IF ( KOPT .LT. 3 ) GOTO 2000
C
        WRITE (OUTPUT,9010) NMS
        WRITE (OUTPUT,9080)
C
        IF ( N1(ISYS) .GT. 0 ) GOTO 1100
```

```
      WRITE (OUTPUT,9090)
      GOTO 2000
C
 1100 HOLDER = 100.0 * RN1SYS
      DO 1200 I = 1,NPARTS
         DX(I) = HOLDER * SYSCRT(I)
 1200 CONTINUE
C
C-----    SORT BY PERCENT OF UNRELIABILITY, THEN PRINT OUT
C
      CALL SORT(DX,KPTR,NPARTS)
C
      KWROT = 0
      DO 1210 I = 1,NPARTS
         KPTRI = KPTR(I)
         IF ( DX(KPTRI) .LE. 0.0 ) GOTO 1210
         WRITE (OUTPUT,9100) KPTRI,TYPE(KPTRI),
     &          ( NAME(TYPE(KPTRI),J), J = 1,4), DX(KPTRI)
         KWROT = KWROT + 1
         IF ( MOD(KWROT,50) .NE. 0) GOTO 1210
         IF ( I .GE. NPARTS )  GOTO 1210
         WRITE (OUTPUT,9010) NMS
         WRITE (OUTPUT,9080)
 1210 CONTINUE
C
C-----    RELIABILITY HIT LIST BY EQUIPMENT TYPES;
C-----    AGGREGATE RELIABILTY HIT DATA BY TYPE OF EQUIPMENT
C
      DO 1220 I = 1,NEQT
         DX(I) = 0.0
 1220 CONTINUE
C
      DO 1230 I = 1,NPARTS
         DX(TYPE(I)) = DX(TYPE(I)) + HOLDER * SYSCRT(I)
 1230 CONTINUE
C
C-----    SORT BY CRITICALITY, THEN PRINT OUT
C
      CALL SORT(DX,KPTR,NEQT)
C
      WRITE (OUTPUT,9010) NMS
      WRITE (OUTPUT,9110)
C
      KWROT = 0
      DO 1240 I = 1,NEQT
         KPTRI = KPTR(I)
         IF ( DX(KPTRI) .LE. 0.0 ) GOTO 1240
         WRITE (OUTPUT,9120) KPTRI,
     &          ( NAME(KPTRI,J), J = 1,4), DX(KPTRI)
         KWROT = KWROT + 1
         IF (MOD(KWROT,50) .NE. 0 ) GOTO 1240
         IF (I .GE. NEQT) GOTO 1240
         WRITE (OUTPUT,9010) NMS
         WRITE (OUTPUT,9110)
 1240 CONTINUE
C
C-----    AVAILABILITY HIT LIST BY EQUIPMENT NUMBERS
C
```

```
      RDENOM = 100.0 / ( SDTIM * NMISS )
      DO 1250 I = 1,NPARTS
         DX(I) = AVCUM(I) * RDENOM
 1250 CONTINUE
C
C-----   SORT BY CRITICALITY, THEN PRINT OUT
C
      CALL SORT(DX,KPTR,NPARTS)
C
      WRITE (OUTPUT,9010) NMS
      WRITE (OUTPUT,9130)
C
      KWROT = 0
      DO 1260 I = 1,NPARTS
         KPTRI = KPTR(I)
         IF ( DX(KPTRI) .LE. 0.0 ) GOTO 1260
         SPARX = AVSPAR(KPTRI) * RDENOM
         WRITE (OUTPUT,9140) KPTRI,TYPE(KPTRI),
     &           ( NAME(TYPE(KPTRI),J), J = 1,4), DX(KPTRI),SPARX
         KWROT = KWROT + 1
         IF ( MOD(KWROT,50) .NE. 0) GOTO 1260
         IF (  I.GE.NPARTS ) GOTO 1260
         WRITE (OUTPUT,9010) NMS
         WRITE (OUTPUT,9130)
 1260 CONTINUE
C
C-----   AVAILABILITY HIT LIST BY EQUIPMENT TYPES;
C-----   AGGREGATE AVAILABILTY HIT DATA BY TYPE OF EQUIPMENT
C
      DO 1270 I = 1,NEQT
          CX(I) = 0.0
          DX(I) = 0.0
 1270 CONTINUE
C
      DO 1290 I = 1,NPARTS
         CX(TYPE(I)) = CX(TYPE(I)) + AVSPAR(I) * RDENOM
         DX(TYPE(I)) = DX(TYPE(I)) +  AVCUM(I) * RDENOM
 1290 CONTINUE
C
C-----   SORT BY CRITICALITY, THEN PRINT OUT
C
      CALL SORT(DX,KPTR,NEQT)
C
      WRITE (OUTPUT,9010) NMS
      WRITE (OUTPUT,9150)
C
      KWROT = 0
      DO 1300 I = 1,NEQT
         KPTRI = KPTR(I)
         IF ( DX(KPTRI) .LE. 0.0 ) GOTO 1300
         WRITE (OUTPUT,9160) KPTRI,
     &           ( NAME(KPTRI,J), J = 1,4), DX(KPTRI),CX(KPTRI)
         KWROT = KWROT + 1
         IF ( MOD(KWROT,50) .NE. 0 ) GOTO 1300
         IF ( I .GE. NEQT ) GOTO 1300
         WRITE (OUTPUT,9010) NMS
         WRITE (OUTPUT,9150)
 1300 CONTINUE
```

```
C
C--------------------
C-----   SPARES REPORT
C--------------------
C
         WRITE (OUTPUT,9170) NMS
C
         DO 1310 I = 1,NEQT
            ICX(I) = 0
            CX(I)  = 0.0
 1310    CONTINUE
C
C-----   AGGREGATE NUMBER OF FAILS (ICX) BY TYPE
C
         DO 1320 I = 1,NPARTS
            ICX(TYPE(I)) = ICX(TYPE(I)) + N1(I)
 1320    CONTINUE
C
         DO 1330 I = 1,NEQT
            DX(I) = NSNOT(I) * RNMISS
            IF ( ICX(I) .GT. 0 )
     &      CX(I) = FLOAT(NOSPRE(I)) / FLOAT(ICX(I))
 1330    CONTINUE
C
         CALL SORT (CX,KPTR,NEQT)
C
         DO 1340 I = 1,NEQT
             KPTRI = KPTR(I)
             SU = N3(KPTRI) * RNMISS
             WRITE (OUTPUT,9180) KPTRI,( NAME(KPTRI,J),J = 1,4),
     &                CX(KPTRI),DX(KPTRI),SU,SPARE1(KPTRI)
 1340    CONTINUE
C
C--------------------------
C-----   REPAIR TIMES REPORT
C--------------------------
C
 2000    IF ( KOPT .LT. 2) GOTO 3000
         IF ( NREP .LE. 0) GOTO 2500
         REPM = RSUM / NREP
         REPSTD = SQRT( ( RSQ - ( RSUM*RSUM / NREP) ) / ( NREP - 1) )
 2500    WRITE (OUTPUT,9190) NMS,REPM,REPSTD,NREP
C
 3000    RETURN
C
C
 9000    FORMAT(/,10X,'BASIC SYSTEM REPORT',/
     1   /,10X,'NUMBER OF MISSIONS:            ',T40,I12,
     2   /,10X,'NUMBER OF SUCCESSFUL MISSIONS:',T40,I12,
     3   /,10X,'RELIABILITY:                   ',T40,F12.4,
     4   /,10X,'AVAILABILITY:                  ',T40,F12.4,
     5   /,10X,'UPTIME/MISSION:                ',T40,F12.2,
     6   /,10X,'DOWNTIME/MISSION:              ',T40,F12.2,
     7   /,10X,'MISSION DURATION :             ',T40,F12.2,
     8   /,10X,'NUMBER OF FAILURES:            ',T40,I12,
     9   /,10X,'MTB MISSION FAILURES:          ',T40,F12.2,
     A   /,10X,'SYSTEM MTTF:                   ',T40,F12.2,
     B   /,10X,'NUMBER OF FAULTS:              ',T40,I12,
```

```
     C   /,10X,'NUMBER OF EVENTS:             ',T40,I12  )
C
 9010    FORMAT(' PAGE BREAK',/,10X,'MODEL IDENT: ',19A4)
C
 9020    FORMAT(/,T10,'MAJOR SUBSYSTEM  REPORT',//,T10,
     &   'SUBSYSTEM  GROUP   ',
     &   'FAILS/MISS    DWNTME/MISS    RELIAB    AVAIL')
C
 9030    FORMAT(T14,A4,T21,I4,T29,F7.3,T38,F13.4,4X,2F9.4)
C
 9040    FORMAT(/,10X,'EQUIPMENT RELIABILITY REPORT',//,
     &   5X,'EQUIPMENT:',17X,'   FAILS/    DWNTME/',/,5X,
     &   'NO    TYPE          NAME          ',
     &   ' MISSION    MISSION     RELIAB     AVAIL ')
C
 9050    FORMAT(1X,2I6,5X,4A4,F7.3,2X,F9.4,2X,2F9.4)
C
 9060    FORMAT(/,10X,'GROUP RELIABILITY REPORT',//,5X,
     &   'GROUP NUMBER     ',
     &   'FAILS/MISS    DWNTME/MISS       RELIAB         AVAIL ')
C
 9070    FORMAT(8X,I6,8X,F7.3,3X,F13.4,2F13.4)
C
 9080    FORMAT(//,15X,'RELIABILITY HIT LIST',
     &   ' BY EQUIPMENT NUMBER',//,T16,'EQUIP NO.',T26,'EQUIP TYPE',
     &   T39,' NAME',T50,' PCT. OF UNRELIABILITY  ')
C
 9090    FORMAT(10X,'NO SYSTEM FAILURES OCCURRED')
C
 9100    FORMAT(10X,2I10,4X,4A4,5X,3F10.4)
C
 9110    FORMAT(//,15X,'RELIABILITY HIT LIST',
     &   ' BY EQUIPMENT TYPE',//,T14,'EQUIP TYPE',T39,
     &   ' NAME',T50,' PCT. OF UNRELIABILITY')
C
 9120    FORMAT(10X,I10,14X,4A4,5X,3F10.4)
C
 9130    FORMAT(//,5X,'AVAILABILITY HIT LIST',
     &   ' BY EQUIPMENT NUMBER',//,T40,'   TOTAL PCT.  SPARES PCT.',/,
     &   T5,'EQUIP NO.',T16,'EQUIP TYPE',T29,
     &   ' NAME',T40,'    UNAVAIL.     UNAVAIL.')
C
 9140    FORMAT(1X,2I10,4X,4A4,2F10.4)
C
 9150    FORMAT(//,5X,'AVAILABILITY HIT LIST',
     &   ' BY EQUIPMENT TYPE',//,T40,'   TOTAL PCT.  SPARES PCT.',/,
     &   T5,'EQUIP TYPE',T29,
     &   ' NAME',T40,'    UNAVAIL.     UNAVAIL.')
C
 9160    FORMAT(1X,I10,14X,4A4,2F10.4)
C
 9170    FORMAT(' PAGE BREAK',/,15X,' ID: ',19A4,//,15X,
     &   'SPARES UNAVAILABILITY & UNRELIABILITY BY EQUIPMENT TYPE',
     &   //,15X,'SORTED ON RATIO OF NO. OF OUTAGES/NO. OF FAILS',//,
     &   T5,'EQUIP TYPE',T18,'NAME',T34,'OUT/FAIL',T45,
     &   'UNREL',T52,'USE/MISS',T63,'ALLOWED')
C
 9180    FORMAT(5X,I5,5X,4A4,2F9.4,F9.4,1X,I10)
```

```
C
 9190   FORMAT(' PAGE BREAK',/,15X,' ID: ',19A4,//,15X,
     &        'REPAIR TIME STATISTICS',//,
     &        T15,'MTBR:',            T35,F9.4,/,
     &        T15,'STD DEV:',         T35,F9.4,/,
     &        T15,'NUMBER OF REPAIRS:',T35,I9)
C
        END
C
```

```
C *********************************************************************
C *********************************************************************
C *********************************************************************
C
        SUBROUTINE REPSURV
C                                                                5/6/92
C----------------------------------------------------------------------
C    PURPOSE:  PRODUCES SURVIVABILITY REPORT
C    CALLED IN: MAIN
C    CALLS TO:
C----------------------------------------------------------------------
C
C-----  FILE ARRAYDIM.INC CONTAINS THE MAXIMUM ARRAY DIMENSIONS AND
C       FILE DECLARE.INC  CONTAINS GLOBAL DECLARATIONS
C
        INCLUDE ARRAYDIM.INC
        INCLUDE DECLARE.INC
C
C----------------------------------------------------------------------
C
      WRITE(OUTPUT, 1000)
 1000 FORMAT(/'THE SURVIVABILITY OF EACH EQUIPMENT (IN PERCENT)')
C
      WRITE(OUTPUT, 1100) NMISS
 1100 FORMAT('FOR ',I4,' SURVIVABILITY TRIALS.'/)
C
      RNMISS = FLOAT(NMISS)
      DO 1200 I = 1, NBLKS
        SURVPCT(I)=(1.0-SNUMFAL(I)/RNMISS)*100.0
 1200 CONTINUE
C
      IF (DISPALL .EQ. 1) GOTO 1399
C
      DO 2000 I = 1, NDISP
        WRITE(OUTPUT, 2900) BLKDISP(I), SURVPCT(BLKDISP(I))
 2900   FORMAT('BLOCK # ',T15,I4,': ',T45,F7.2,' PERCENT.')
 2000 CONTINUE
      GOTO 9000
C
 1399 DO 1300 I = 1, NPARTS
        WRITE(OUTPUT, 1400) I, SURVPCT(I)
 1400   FORMAT(' EQUIPMENT # ',T15,I4,': ',T45,F7.2,' PERCENT.')
 1300 CONTINUE
C
      DO 1500 I = LOWGRP, NBLKS
        WRITE(OUTPUT, 1600) I, SURVPCT(I)
 1600   FORMAT(' GROUP # ',T15,I4,': ',T45,F7.2,' PERCENT.')
 1500 CONTINUE
C
      WRITE(OUTPUT, 2100) COSTTL
 2100 FORMAT(/'THE COST OF THE SYSTEM IS $',F12.2/)
C
      IF (DISPALL .EQ. 1) GOTO 2399
C
      DO 2700 I = 1, NDISP
      WRITE(OUTPUT, 2800) BLKDISP(I),(10000.*SURVPCT(BLKDISP(I)))/COSTTL
 2800 FORMAT('BLOCK # ',T15,I4,'SURV COST INDEX: ',T45,F6.1/)
 2700 CONTINUE
```

```
      GOTO 9000
C
 2399 DO 2300 I = 1, NPARTS
        WRITE(OUTPUT, 2400) I, (10000.*SURVPCT(I))/COSTTL
 2400   FORMAT(' EQUIPMENT # ',T15,I4,'SURV COST INDEX: ',T45,F6.1/)
 2300 CONTINUE
C
      DO 2500 I = LOWGRP, NBLKS
        WRITE(OUTPUT, 2600) I, (10000.*SURVPCT(I))/COSTTL
 2600   FORMAT(' GROUP # ',T15,I4,'SURV COST INDEX: ',T45,F6.1/)
 2500 CONTINUE
C
 9000 RETURN
C
      END
C
```

```
C ************************************************************************
C ************************************************************************
C ************************************************************************
C
      SUBROUTINE SET
C                                                                12/04/89
C-----------------------------------------------------------------------
C   PURPOSE:   CHANGES THE SET OF SUBSYSTEMS INCLUDED IN THE SYSTEM,
C              AT MISSION PHASE BOUNDARIES.
C   CALLED IN: INITAL, PCHANG
C   CALLS TO:  STAT
C-----------------------------------------------------------------------
C
C-----    FILE ARRAYDIM.INC CONTAINS THE MAXIMUM ARRAY DIMENSIONS AND
C         FILE DECLARE.INC  CONTAINS GLOBAL DECLARATIONS
C
      INCLUDE ARRAYDIM.INC
      INCLUDE DECLARE.INC
C
C-----------------------------------------------------------------------
C
C
C-----    HANDLE VARIABLE REQUIRMENT GROUPS, IF APPLICABLE
C
      IF ( NGRCHG.LE.0 ) GOTO 1300
C
      DO 1050 I = 1,NGRCHG
         IGRPCH = GRPCHG(I)
         IGROUP = IGRPCH - LOWGRP +1
C
C-----     CHANGE NUMBER NEEDED BY GROUP FOR CURRENT PHASE TYPE
C
         NONEED(IGROUP) = GRPREQ(I,PHASE)
C
C-----     CHECK GROUP STATUS
C
         IF ( UP(IGRPCH) .NE. 0 ) GOTO 1025
C
C-----     CASE 1: GROUP UP -> REMAINS UP
C
         IF ( NOUP(IGROUP) .GE. NONEED(IGROUP) ) GOTO 1050
C
C-----     CASE 2: GROUP UP -> BROUGHT DOWN BY NEW REQUIREMENTS
C
         CALL STAT(IGRPCH,1)
         GOTO 1050
C
C-----     CASE 4: GROUP DOWN -> REMAINS DOWN
C
 1025    IF ( NOUP(IGROUP) .LT. NONEED(IGROUP) ) GOTO 1050
C
C-----     CASE 3: GROUP DOWN -> BROUGHT UP BY NEW REQUIREMENTS
C
         CALL STAT(IGRPCH,0)
C
 1050 CONTINUE
 1300 CONTINUE
C
```

```
      RETURN
      END
C
```

```
C ************************************************************************
C ************************************************************************
C ************************************************************************
C
      SUBROUTINE SORT (B,PT,N)
C                                                                02/29/88
C-----------------------------------------------------------------------
C    PURPOSE:   SORTS THE INDEX LIST PT(*) TO REFLECT A DESCENDING
C               ORDER IN THE UNMODIFIED LIST B(*)
C    CALLED IN: REPORT
C-----------------------------------------------------------------------
C
      INTEGER   PT
      DIMENSION B(N), PT(N)
C
C-----------------------------------------------------------------------
C
C-----   CREATE INITIAL PT(*) LIST
C
      DO 10 I = 1,N
         PT(I) = I
   10 CONTINUE
C
C-----   FORM THE HEAP
C
      DO 30 I = 2,N
         JL = I
         J  = PT(I)
   20    KL = JL / 2
         K  = PT(KL)
         IF ( B(J).GT.B(K) ) GOTO 30
         PT(KL) = J
         PT(JL) = K
         JL = KL
         J  = PT(JL)
         IF ( JL.GT.1 ) GOTO 20
   30 CONTINUE
C
      I = N
  100 IF ( I.LT.1 ) RETURN
C
C-----   PUT 1-ST ELEMENT OF HEAP AT BEGINNING OF SORTED ELEMENT SECTION
C
      IT    = PT(1)
      PT(1) = PT(I)
      PT(I) = IT
      I  = I - 1
      JL = 1
      KL = 2
      J = PT(1)
      K = PT(2)
C
C-----   PUSH NEW TOP MEMBER BACK INTO PROPER PLACE IN HEAP
C
  110 IF ( KL. GT. I ) GOTO 100
         KL1 = KL + 1
         KP1 = PT(KL1)
         IF ( KL1.GT.I .OR. B(K).LT.B(KP1) ) GOTO 120
```

```
            K  = KP1
            KL = KL1
  120       IF ( B(J) .LE. B(K) ) GOTO 100
            PT(JL) = K
            PT(KL) = J
            JL = KL
            KL = 2 * JL
            IF ( KL .GT. I) GOTO 100
            J = PT(JL)
            K = PT(KL)
         GOTO 110
C
         END
C
```

```
C ***********************************************************************
C ***********************************************************************
C ***********************************************************************
C
      SUBROUTINE SSORT (A,B,N)
C                                                               02/29/88
C-----------------------------------------------------------------------
C    PURPOSE:    SORTS LIST B(*) IN ASCENDING ORDER AND
C                PERFORMS SAME INTERCHANGES IN LIST A(*).
C    CALLED IN: READIT
C-----------------------------------------------------------------------
C
      INTEGER   A,    B
      DIMENSION A(N), B(N)
C
C-----------------------------------------------------------------------
C
C-----   INITIALIZE INTERVAL AT FULL RANGE
C
      JMP = N
C
C-----   NO INTERCHANGES ENCOUNTERED; HALVE THE INTERVAL
C
   10 JMP = JMP / 2
      IF ( JMP.LT.1 ) RETURN
      IEND = N - JMP
C
C-----   INTERCHANGES ENCOUNTERED; RESET FLAG; MAINTAIN INTERVAL
C
   20 ISW  = 0
C
      DO 100 I = 1,IEND
         J = I + JMP
         IF ( B(I).LE.B(J) ) GOTO 100
         ITEMP = B(I)
         B(I)  = B(J)
         B(J)  = ITEMP
         ITEMP = A(I)
         A(I)  = A(J)
         A(J)  = ITEMP
         ISW   = 1
  100 CONTINUE
      IF ( ISW ) 10,10,20
C
      END
C
```

```
C ***********************************************************************
C ***********************************************************************
C ***********************************************************************
C
      SUBROUTINE STAT (BLK,RF)
C                                                                03/07/88
C-----------------------------------------------------------------------
C   PURPOSE:    EVALUATES THE CHANGE IN THE STRUCTURE FUNCTION. THE
C               STRUCTURE FUNCTION IS REPRESENTED AS A NETWORK, PARTS
C               BEING THE SOURCES AND THE SYSTEM NODE BEING THE SINK.
C               AFTER A CHANGE OF STATE IN THE BLOCK POINTED TO BY "BLK"
C               THIS ROUTINE IS CALLED. THE EFFECT OF THE CHANGE IS
C               TRACED THROUGH THE NETWORK AND STATISTICS AT EACH BLOCK
C               ENCOUNTERED ARE UPDATED.
C   CALLED IN: MAIN, SET
C   CALLS TO:  ERRDKE
C   NOTE:       IN THIS SUBROUTINE THE STACK POINTER IS NOW TRACING
C               FROM THE BOTTOM UP.  IF A PIECE OF EQUIPMENT IS SHARED
C               BY SEVERAL GROUPS (RESOURCE SHARING), THE POINTER
C               PROVIDES BOTH HORIZONTAL AND VERTICAL TRACING.
C               IT STARTS AT THE EQUIPMENT LEVEL, MOVES TO THE HIGHEST
C               VERTICAL LEVE AFFECTED BY THE CURRENT FAILURE OR REPAIR
C               EVENT, THEN MOVES HORIZONTALLY TO THE NEXT GROUP AFFECTED.
C               WHEN THERE IS NO BOTTOM-UP RESOURCE SHARING, THE STACK
C               POINTER IS SUPERFLUOUS IN THIS SUBROUTINE.
C-----------------------------------------------------------------------
C
C-----   FILE ARRAYDIM.INC CONTAINS THE ARRAY DIMENSIONS AND
C        FILE DECLARE.INC CONTAINS GLOBAL DECLARATIONS
C
      INCLUDE ARRAYDIM.INC
      INCLUDE DECLARE.INC
C
C-----   LOCAL DECLARATIONS
C
      INTEGER      RF,     SP
C
C        END OF LOCAL DECLARATIONS
C
C-----------------------------------------------------------------------
C
C-----   INITIALIZE THE STACK POINTER
C
      SP = 0
C
C-----   EVENT TYPE: RF = 0/1 -> REPAIR / FAILURE OF BLOCK
C
 1000 IF ( RF.EQ.0 ) GOTO 1100
C
C-----   FAIL THIS BLOCK - START DOWNTIME STATISTICS
C
      UP(BLK)      = 1
      DWNTME(BLK)  = TIME
      NUMFAL(BLK)  = NUMFAL(BLK)+1
      GOTO 1200
C
C-----   REPAIR THIS BLOCK - CLOSE OFF DOWNTIME STATISTICS
C
```

```
 1100   UP(BLK)      = 0
        CDWNTM(BLK) = CDWNTM(BLK) + ( TIME - DWNTME(BLK) )
        DWNTME(BLK) = TIME
C
C-----  START TRACING THE EFECT OF THE EVENT INVOLVING THIS
C       BLOCK ON THE REST OF THE SYSTEM
C
C-----  SET ARC INDICES (NOTE THE USE OF FSTAR, WHEREAS RSTAR
C       IS USED IN BAKST1,2,3)
C
 1200   IBEG  = FSTAR(BLK)
        IEND  = FSTAR(BLK+1) - 1
        IND   = IBEG
C
 1300   IF ( IND .GT. IEND ) GOTO 1600
        TBLK   = ARC(IND)
        TBLKGR = TBLK - LOWGRP + 1
        IF ( TBLK .LT. 0 )   GOTO 1700
C
C-----  THE PREVIOUS LINE APPEARS TO BE A RELIC FROM GSTAT!
C
        IF ( RF .EQ. 0)    GOTO 1400
C
        NOUP(TBLKGR) = NOUP(TBLKGR) - 1
        IF (   UP(TBLK) .NE. 0 )                GOTO 1700
        IF ( NOUP(TBLKGR) .GE. NONEED(TBLKGR) ) GOTO 1700
        GOTO 1500
C
 1400   NOUP(TBLKGR) = NOUP(TBLKGR) + 1
        IF (   UP(TBLK) .EQ. 0 )                GOTO 1700
        IF ( NOUP(TBLKGR) .LT. NONEED(TBLKGR) ) GOTO 1700
C
 1500   SP = SP + 4
        IF ( SP.GT.LEN12 ) CALL ERRKDE (8,SP,LEN12)
        STAK(SP-3) = BLK
        STAK(SP-2) = IBEG
        STAK(SP-1) = IEND
        STAK(SP)   = IND
        BLK = TBLK
        GOTO 1000
C
 1600   IF ( SP .EQ. 0 ) RETURN
C
        TBLK = BLK
        IND  = STAK(SP)
        IEND = STAK(SP-1)
        IBEG = STAK(SP-2)
        BLK  = STAK(SP-3)
        SP   = SP - 4
C
 1700   IND = IND + 1
        GOTO 1300
C
        END
C
```

```
C ***********************************************************************
C ***********************************************************************
C ***********************************************************************
C
      SUBROUTINE SYS_STATS
C                                                               11/21/89
C-----------------------------------------------------------------------
C    PURPOSE: IN THIS SUBROUTINE WE ESTABLISH SYSTEM STATUS TRANSITION
C             CASES AND HANDLE ACCORDINGLY.
C    CALLED IN: MAIN
C    CALLS TO:  BAKST1, BAKST2, BAKST3
C-----------------------------------------------------------------------
C
C-----    FILE ARRAYDIM.INC CONTAINS THE MAXIMUM ARRAY DIMENSIONS AND
C         FILE DECLARE.INC  CONTAINS GLOBAL DECLARATIONS
C
      INCLUDE ARRAYDIM.INC
      INCLUDE DECLARE.INC
C
C----------------------------------------------------------------------
C
C----  WE HAVE TO PUT IN THE CORRECT FILE CALLS TO THE
C      COMMON BLOCKS, ETC.
C
C-------------------------------------------------------
C  CASE     SYSTEM STATUS TRANSITION HANDLING:
C
C   1       UP           -> UP:   DO =>  NOTHING
C   2       UP           -> DOWN: DO =>  BAKST1 -> BAKST2
C   3       DOWN         -> UP:   DO =>  BAKST3
C   4       DOWN         -> DOWN: DO =>  BAKST3 -> BAKST1
C------------------------------------------------------
C
      IF ( ISAVE .NE. 0 ) GOTO 1100
C
C-----    CASE 1: SYSTEM HAD BEEN UP BEFORE THIS FAILURE (ISAVE=0)
C                 AND WAS NOT BROUGHT DOWN BY IT (UP(ISYS)=0)
C
      IF ( UP(ISYS) .EQ. 0) GOTO 2000
C
C-----    CASE 2: SYSTEM HAD BEEN UP BEFORE THIS FAILURE (ISAVE=0)
C                 AND WAS BROUGHT DOWN BY IT (UP(ISYS) .NE. 0)
C
      CALL BAKST1
      CALL BAKST2
C
C-----    GENERATE FIRST FAILURE TIME FILE (IF APPLICABLE AND
C         REQUIRED). THIS FILE IS UPDATED ONLY AT THE FIRST SYSTEM
C         FAILURE WHICH OCCURS IN A MISSION.  SUBSEQUENT FAILURES
C         WITHIN THE SAME MISSION ARE NOT COUNTED FOR THIS PURPOSE.
C
      IF ( NUMFAL(ISYS) .GT. 1 ) GOTO 2000
      NFT  = NFT + 1
C
      IF ( KOPT.LT.5 )          GOTO 2000
      GOTO 2000
C
C-----    CASE 4: SYSTEM HAD BEEN DOWN BEFORE THIS FAILURE (ISAVE.NE.0)
```

```
C              AND REMAINED DOWN FOLLOWING IT (UP(ISYS) .NE. 0)
C-----         MISSION CONTINUES EVEN THOUGH IT IS ABORTED STATUS.
C-----         SEQUENCE BAKST3-BAKST1 IS USED TO TRACK THE CHANGE
C              IN SYSTEM CONFIGURATION.
C
 1100   CALL BAKST3
        IF ( UP(ISYS) .NE. 0) CALL BAKST1
        GOTO 2000
 2000   CONTINUE
C
        RETURN
        END
C
```

```
C *********************************************************************
C *********************************************************************
C *********************************************************************
C
      SUBROUTINE ZZERO
C                                                              4/23/92
C-------------------------------------------------------------------
C    PURPOSE:   ZEROES OUT ALL THE ARRAYS NOT RESET IN INITAL.
C    CALLED IN: MAIN
C-------------------------------------------------------------------
C
C-----   FILE ARRAYDIM.INC CONTAINS THE MAXIMUM ARRAY DIMENSIONS AND
C        FILE DECLARE.INC  CONTAINS GLOBAL DECLARATIONS
C
      INCLUDE ARRAYDIM.INC
      INCLUDE DECLARE.INC
C
C-------------------------------------------------------------------
C
      DO 100 I = 1,LEN07
         IVDC(I)    = 0
         LAM(I)     = 0.0
C
         DO 50 J = 1,LEN02
            MU(I,J) = 0.0
  50     CONTINUE
C
         N3(I)      = 0
         NOSPRE(I)  = 0
         NSNOT(I)   = 0
         PRI(I)     = 0
         SPARE1(I)  = 0
 100  CONTINUE
C
      DO 200 I = 1,LEN08
         AVCUM(I)   = 0
         AVSPAR(I)  = 0
         SYSCRT(I)  = 0.0
         TYPE(I)    = 0
      EQTYPE(I) = 0
      RADIUS(I) = 0.0
      RADTOHIT(I) = 0.0
      HITDAM(I) = 0
      HITDATA(I) = 0.0
      EQTEMP(I) = 0.0
 200  CONTINUE
C
      DO 300 I = 1,LEN10
         ARC(I)   = 0
         RARC(I)  = 0
 300  CONTINUE
C
      DO 500 I = 1,LEN05
         PHZ(I)     = 0
         PHTIME(I)  = 0.0
 500  CONTINUE
C
      DO 600 I = 1,LEN02
```

```
            PHSRV(I) = 0
  600     CONTINUE
C
          DO 800 I = 1,LEN06
            GRPCHG(I) = 0
  800     CONTINUE
C
          DO 900 I = 1,LEN04
            DO 850 J = 1,LEN02
              VDC(I,J) = 0.0
  850       CONTINUE
  900     CONTINUE
C
          DO 1000 I = 1,3
            FQ(I) = 0
            SQ(I) = 0
 1000     CONTINUE
C
          DO 1100 I = 1,LEN11
             PRT(I) = 0
             RMS(I) = 0
 1100     CONTINUE
C
          DO 1200 I = 1,LEN12
            STAK(I)   = 0
 1200     CONTINUE
C
          DO 1300 I = 1,LEN13
            FSTAR(I)  = 0
            RSTAR(I)  = 0
 1300     CONTINUE
C
          DO 1400 I = 1,LEN14
            NONEED(I) = 0
            NOOFF(I)  = 0
            TEMPUP(I) = 0
 1400     CONTINUE
C
          DO 1500 I = 1,LEN15
            SST1(I)   = 0.0
            T1(I)     = 0.0
            UP(I) = 0
          SNUMFAL(I) = 0.0
 1500     CONTINUE
C
          DO 1600 I = 1,LEN16
            N1(I)     = 0
            N2(I)     = 0
 1600     CONTINUE
C
          DO 1700 I = 1,19
            NMS(I) = 0
 1700     CONTINUE
C
          DO 1800 I = 1,LEN08
            DO 1850 J = 1,9
              EQDATA(I,J) = 0.0
 1850       CONTINUE
```

```
 1800 CONTINUE
C
      DO 1900 I = 1, LEN17
         SOURCE(I) = 0
 1900 CONTINUE
C
      DO 2000 I = 1, LEN18
         LOAD(I) = 0
 2000 CONTINUE
C
      DO 2100 I = 1, LEN19
         BREAKER(I) = 0
 2100 CONTINUE
C
      DO 2200 I = 1, LEN20
         CABLE(I) = 0
 2200 CONTINUE
C
      DO 2300 I = 1, LEN21
         CABINET(I) = 0
 2300 CONTINUE
C
      DO 2400 I = 1, LEN22
         BUSTIE(I) = 0
 2400 CONTINUE
C
      DO 2500 I = 1, LEN23
         ABT(I) = 0
 2500 CONTINUE
C
      DO 2600 I = 1, LEN24
         ALTSRC(I) = 0
 2600 CONTINUE
C
      DO 2700 I = 1, ((LEN18-LEN23)+2*LEN23)*LEN17
        DO 2710 J = 1, LEN08
          PWRPATH(I,J) = 0
          SRCPATH(I,J) = 0
 2710   CONTINUE
 2700 CONTINUE
C
         RETURN
C
         END
```

Appendix D. Include Files

```
C -----  FILE ARRAYDIM.INC
C
C
C-----    THE FOLLOWING ARRAY DIMENSIONS ARE CUSTOMIZED
C         FOR << ELECTRICAL DISTRIBUTION SYSTEM SURVIVABILITY 5/6/92 >>
C
C      DIMENSION   DESCRIPTION
C
C         LEN01    = NUMBER OF PARTS PLUS TWO (NP2)
C         LEN02    = NUMBER OF PHASE TYPES (NPT)
C         LEN03    = NUMBER OF SUBSYSTEMS (NSSYS)
C         LEN04    = MAXIMUM OF VARIABLE REQUIREMENTS (NVDC,NCOV,NISOL)
C         LEN05    = NUMBER OF PHASES PLUS FIVE (NPHASE + 5)
C         LEN06    = NUMBER OF GROUPS WITH VARYING
C                       REQUIREMENTS (NGRCHG)
C         LEN07    = NUMBER OF EQUIPMENT TYPES (NEQT)
C         LEN08    = NUMBER OF PARTS (NPARTS)
C         LEN09    = MAXIMUM NUMBER OF FIRST SYSTEM FAILURES
C                       OVER ALL MISSIONS (NMISS)
C         LEN10    = NUMBER OF ARCS (NARC), I.E. GROUP-TO-
C                       MEMBER CONNECTIONS
C         LEN11    = ALLOWABLE NUMBER OF ITEMS IN QUEUE
C                       (MAXLST)
C         LEN12    = FOUR TIMES THE ALLOWABLE NUMBER OF
C                       LEVELS IN NETWORK
C         LEN13    = NUMBER OF BLOCKS PLUS ONE (NBLKS + 1)
C         LEN14    = NUMBER OF GROUPS (NGROUP)
C         LEN15    = NUMBER OF BLOCKS
C                       (NBLKS = LOWGRP - 1 + NGROUP)
C         LEN16    = MAXIMUM OF NUMBER OF ARCS (NARC) AND
C                       NUMBER OF BLOCKS (NBLKS)
C         LEN17    = NUMBER OF POWER SOURCES (NUMSRC)
C         LEN18    = NUMBER OF LOADS (NUMLD)
C         LEN19    = NUMBER OF BREAKERS (NUMBKR)
C         LEN20    = NUMBER OF CABLES (NUMCBL)
C         LEN21    = NUMBER OF CABINETS (NUMCAB)
C         LEN22    = NUMBER OF BUS TIES (NUMBT)
C         LEN23    = NUMBER OF ABT (NUMABT)
C         LEN24    = NUMBER OF ALTERNATE DC SOURCES (NUMALT)
C
          PARAMETER   ( LEN01 =    98, LEN02 =     1, LEN03 =     1,
     1                  LEN04 =     1, LEN05 =    10, LEN06 =     1,
     2                  LEN07 =     8, LEN08 =    96, LEN09 =     1,
     3                  LEN10 =   166, LEN11 =  5000, LEN12 =    60,
     4                  LEN13 =   156, LEN14 =    56, LEN15 =   155,
     5                  LEN16 =   166, LEN17 =     3, LEN18 =     6,
     6                  LEN19 =    19, LEN20 =    58, LEN21 =     4,
     7                  LEN22 =     4, LEN23 =     2, LEN24 =     1)
C
```

```
C         DECLARE.INC
C         ---------------------
C         ------  5/6/92  ------
C         ---------------------
C          GLOBAL DECLARATIONS:
C         ---------------------
C
C        INCLUDING NEW VARIABLES FOR SURVIVABILITY ANALYSIS.
C
          INTEGER     ARC, FSTAR,GRPCHG,GRPREQ, PHSRV, PHZ,IVDC,KPTR,
     1                   PRED,   PRI,   PRT,  RARC,   RMS, RSTAR,SPARE1,
     2                SPARES,  STAK,  SUCC,TEMPUP,  TYPE, UP
          INTEGER     FQ, PHASE,   PHI,REMSTG, SEED1, SEED2, SEED3,
     1                   SEED4,      SQ, SRVBLK, SRVOP, IND, SRCPATH
          INTEGER     AUX1,  AUX2,  AUX3,CONSOL,OUTPUT, EQTYPE
          INTEGER     BLK,FQFRST,FQLAST, FQNUM,SQFRST,SQLAST, SQNUM,
     1                   TBLK, HITDAM, ALTSRC, DISPALL, BLKDISP, PWRPATH
C
       INTEGER        SURVIVE, COSTCODE, EQCODE, NUMSRC, NUMLD,NUMALT,
     1                   NUMBKR, NUMCBL, NUMCAB, NUMBT, NUMABT, EQUIPNO,
     2                SOURCE, LOAD, BREAKER, CABINET, ABT, BUSTIE, CABLE
C
          REAL*4      LAM, MU
          REAL*4      LLIM, NSDEV, CBLENGTH
C
          REAL*8      RATE
C
          LOGICAL     DONE
C
C---------------------------------------------------------------------
C
          DIMENSION     ARC(LEN10), AVCUM(LEN08),AVSPAR(LEN08),
     1                CDWNTM(LEN15),     CX(LEN07),DWNTME(LEN15),
     2                    DX(LEN08), FSTAR(LEN13),GRPCHG(LEN06),
     3                GRPREQ(LEN06,LEN02),         IADFLG(LEN08),
     4                    ICX(LEN07)
C
          DIMENSION       IVDC(len07), KPTR(len08), LAM(LEN07),
     1                 MU(LEN07,LEN02),
     2                    N1(LEN16),     N2(LEN16),     N3(LEN07),
     3                 NAME(LEN07,4),              NONEED(LEN14),
     4                    NOOFF(LEN14),NOSPRE(LEN07),  NOUP(LEN14)
C
          DIMENSION    NSNOT(LEN07),NUMFAL(LEN15),       P(LEN01),
     1                    PHSRV(LEN02),
     2                 PHTIME(LEN05),    PHZ(LEN05),   PRED(LEN11),
     3                    PRI(LEN07),    PRT(LEN11),   RARC(LEN10),
     4                 RMS(LEN11), RSTAR(LEN13),SPARE1(LEN07)
C
          DIMENSION    SPARES(LEN07),  SST1(LEN15),  STAK(LEN11),
     1                    SUCC(LEN11),SYSCRT(LEN08),SYSDWN(LEN15),
     2                 T1(LEN15),TEMPUP(LEN14),
     3                    TYPE(LEN08),    UP(LEN15),
     4                 VDC(LEN04,LEN02)
C
          DIMENSION    FQ(  3  ),    NMS(  19 ),    SQ(  3  )
C
      DIMENSION        EQTYPE(LEN08), EQDATA(LEN08, 9), RADIUS(LEN08),
```

```
     1                 RADTOHIT(LEN08), HITDAM(LEN08), EQTEMP(LEN08),
     2              HITDATA(LEN08), BLKDISP(LEN08)
C
      DIMENSION     SOURCE(LEN17), LOAD(LEN18), BREAKER(LEN19),
     1                 CABLE(LEN20), CABINET(LEN21), BUSTIE(LEN22),
     2              ABT(LEN23), ALTSRC(LEN24)
C
      DIMENSION     SURVPCT(LEN15), SNUMFAL(LEN15),
     1                 GENLOAD(LEN17,LEN18), SRCLD(LEN17),
     2              PLLCHK(LEN18), LOADWN(LEN18), OVRLD(LEN17),
     3                 PWRPATH(((LEN18-LEN23)+2*LEN23)*LEN17, LEN08),
     4              SRCPATH(((LEN18-LEN23)+2*LEN23)*LEN17, LEN08)
C
         COMMON /       /  ARC, AVCUM,AVSPAR,CDWNTM,CX,DWNTME,
     1           NUMLD, DX, FSTAR,GRPCHG,GRPREQ,IADFLG,
     2           IVDC,  KPTR,   LAM, SOURCE, EQTYPE,
     3           MU,    N1,     N2,     N3,  NAME, NOOFF,
     4           NONEED,NOSPRE,  NOUP, NSNOT, NUMFAL, P,
     5           PHSRV,PHTIME,   PHZ,  PRED,   PRI,  HITDATA,
     6           PRT,  RARC,    RMS, RSTAR, SPARE1, SPARES,
     7           SST1,  STAK,   SUCC,SYSCRT,SYSDWN,    T1,
     8           TYPE,    UP,   VDC, EQDATA
C
         COMMON /       /  NUMCBL, NUMCAB, NUMBT, TEMPUP,
     1           ABT, CABLE, CABINET, BUSTIE, ICX,
     2           NUMBKR,  BREAKER, HITDAM, NUMSRC, NDISP,
     3           HITX, HITY, HITZ, CX1, CY1, CZ1,
     4           CX2, CY2, CZ2, SNUMFAL, SURVIVE, COSTCODE,
     5           LOAD, RDAM , EQCODE, RADIUS, BLKDISP,
     6           DISPALL, GENLOAD, SRCLD, PLLCHK, LOADWN, OVRLD
C
         COMMON /       /  ALTSRC, PWRPATH, COSTTL, COSTEQ, CBLENGTH,
     1           SRCPATH, MNO, NUMABT
C
         COMMON / IVAR /    FQ, IQEND,  ISYS,  KOPT,LOWGRP,MAXLST,
     1           MISSPR,  NBAD, NBLKS,  NEQT,NEVENT,NGRCHG,
     2           NGROUP,NINLST, NMISS,   NMS,   NP1,    NP2,
     3           NPARTS,NPHASE,   NPT,  NREP, NSSYS, NSTAG,
     4           NTFALS,  NVDC, PHASE,   PHI,REMSTG, SEED1,
     5           SEED2, SEED3,     SQ, SRVBLK, SRVOP,    NFT,
     6           ISAVE, SEED4,IUNLIM, IPRI
C
         COMMON / RVAR /  DONE,DURMIS,  LLIM,OPNTME, NSDEV,  RATE,
     1           RSQ,  RSUM, RTIME, SRVMU,  TIME
C
         COMMON / UNIT /  INPUT,OUTPUT,CONSOL,AUX1,AUX2,AUX3,MRK1,MRK2
C
C
         EQUIVALENCE  (FQ(1),FQFRST), (FQ(2),FQLAST), (FQ(3),FQNUM),
     1                (SQ(1),SQFRST), (SQ(2),SQLAST), (SQ(3),SQNUM)
C
C       --------------------------
C        END OF GLOBAL DECLARATIONS
C       --------------------------
```